
Aries Cloud Agent Python Documentation

Nicholas Rempel, Andrew Whitehead, Ian Costanzo, Stephen Klur

Mar 09, 2020

Aries Cloud Agent Python - Modules

1	aries_cloudagent package	3
1.1	Subpackages	3
1.1.1	aries_cloudagent.admin package	3
1.1.2	aries_cloudagent.cache package	6
1.1.3	aries_cloudagent.commands package	7
1.1.4	aries_cloudagent.config package	9
1.1.5	aries_cloudagent.core package	18
1.1.6	aries_cloudagent.holder package	23
1.1.7	aries_cloudagent.issuer package	25
1.1.8	aries_cloudagent.ledger package	25
1.1.9	aries_cloudagent.messaging package	32
1.1.10	aries_cloudagent.storage package	63
1.1.11	aries_cloudagent.transport package	71
1.1.12	aries_cloudagent.utils package	89
1.1.13	aries_cloudagent.verifier package	95
1.1.14	aries_cloudagent.wallet package	96
1.2	Submodules	113
1.3	aries_cloudagent.version module	113
2	aries_cloudagent.connections package	115
2.1	Subpackages	115
2.1.1	aries_cloudagent.connections.models package	115
3	aries_cloudagent.protocols package	131
3.1	Subpackages	131
3.1.1	aries_cloudagent.protocols.actionmenu package	131
3.1.2	aries_cloudagent.protocols.basicmessage package	142
3.1.3	aries_cloudagent.protocols.connections package	144
3.1.4	aries_cloudagent.protocols.credentials package	159
3.1.5	aries_cloudagent.protocols.discovery package	176
3.1.6	aries_cloudagent.protocols.introduction package	179
3.1.7	aries_cloudagent.protocols.issue_credential package	185
3.1.8	aries_cloudagent.protocols.present_proof package	207
3.1.9	aries_cloudagent.protocols.presentations package	228
3.1.10	aries_cloudagent.protocols.problem_report package	237
3.1.11	aries_cloudagent.protocols.routing package	239
3.1.12	aries_cloudagent.protocols.trustping package	251

4 Indices and tables	255
Python Module Index	257
Index	263

Hyperledger Aries Cloud Agent Python (ACA-Py) is a foundation for building decentralized identity applications and services running in non-mobile environments.

This is the Read The Docs site for the Hyperledger [Aries Cloud Agent Python](#). This site contains only the ACA-Py docstrings documentation extracted from the Python Code. For other documentation, please consult the links in the Readme for the [ACA-Py GitHub Repo](#).

If you are getting started with verifiable credentials or Aries, we recommend that you start with this [verifiable credentials and agents getting started guide](#).

Want to quick overview of the deployment model for ACA-Py? See [this document](#).

To investigate the code, use search or click the package links in the left menu to drill into the modules, subpackages and submodules that make up ACA-Py.

Developers that are interested in what DIDComm protocols are supported in ACA-Py should take a look at the [protocols](#) package. These should align with the corresponding [aries-rfcs protocols](#). Decorators defined in aries-rfcs and implemented in ACA-Py can be found [here](#). Some general purpose subpackages that might be of interest include [wallet](#) and [storage](#). For those agents playing different roles in a verifiable credential exchange, take a look at the [issuer](#), [holder](#) and [verifier](#) packages.

Please see the [ACA-Py Contribution guidelines](#) for how to contribute to ACA-Py, including for how to submit issues about ACA-Py.

aries_cloudagent package

Aries Cloud Agent.

1.1 Subpackages

1.1.1 aries_cloudagent.admin package

Submodules

aries_cloudagent.admin.base_server module

Abstract admin server interface.

```
class aries_cloudagent.admin.base_server.BaseAdminServer
    Bases: abc.ABC
```

Admin HTTP server class.

```
add_webhook_target (target_url: str, topic_filter: Sequence[str] = None, max_attempts: int =
                    None)
    Add a webhook target.
```

```
remove_webhook_target (target_url: str)
    Remove a webhook target.
```

```
send_webhook (topic: str, payload: dict)
    Add a webhook to the queue, to send to all registered targets.
```

```
start () → None
    Start the webserver.
```

Raises AdminSetupError – If there was an error starting the webserver

```
stop () → None
    Stop the webserver.
```

aries_cloudagent.admin.error module

Admin error classes.

exception aries_cloudagent.admin.error.**AdminError**(*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.core.error.BaseError*

Base class for Admin-related errors.

exception aries_cloudagent.admin.error.**AdminSetupError**(*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.admin.error.AdminError*

Admin server setup or configuration error.

aries_cloudagent.admin.server module

Admin server classes.

class aries_cloudagent.admin.server.**AdminModulesSchema**(*args, **kwargs)

Bases: *sphinx.ext.autodoc.importer._MockObject*

Schema for the modules endpoint.

result

Used by autodoc_mock_imports.

class aries_cloudagent.admin.server.**AdminResponder**(context: *aries_cloudagent.config.injection_context.InjectionContext*, send: *Coroutine[T_co, T_contra, V_co]*, webhook: *Coroutine[T_co, T_contra, V_co]*, **kwargs)

Bases: *aries_cloudagent.messaging.responder.BaseResponder*

Handle outgoing messages from message handlers.

send_outbound(message: *aries_cloudagent.transport.outbound.message.OutboundMessage*)

Send outbound message.

Parameters **message** – The *OutboundMessage* to be sent

send_webhook(topic: str, payload: dict)

Dispatch a webhook.

Parameters

- **topic** – the webhook topic identifier
- **payload** – the webhook payload value

class aries_cloudagent.admin.server.**AdminServer**(host: str, port: int, context: *aries_cloudagent.config.injection_context.InjectionContext*, outbound_message_router: *Coroutine[T_co, T_contra, V_co]*, webhook_router: *Callable*, task_queue: *aries_cloudagent.utils.task_queue.TaskQueue* = None, conductor_stats: *Coroutine[T_co, T_contra, V_co]* = None)

Bases: *aries_cloudagent.admin.base_server.BaseAdminServer*

Admin HTTP server class.

add_webhook_target (*target_url: str, topic_filter: Sequence[str] = None, max_attempts: int = None*)

Add a webhook target.

make_application () → *<sphinx.ext.autodoc.importer._MockObject object at 0x7f9e766535c0>*

Get the aiohttp application instance.

on_startup (*app: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e766535c0>*)

Perform webserver startup actions.

plugins_handler (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e766535c0>*)

Request handler for the loaded plugins list.

Parameters **request** – aiohttp request object

Returns The module list response

redirect_handler (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e766535c0>*)

Perform redirect to documentation.

remove_webhook_target (*target_url: str*)

Remove a webhook target.

send_webhook (*topic: str, payload: dict*)

Add a webhook to the queue, to send to all registered targets.

start () → None

Start the webserver.

Raises `AdminSetupError` – If there was an error starting the webserver

status_handler (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e766535c0>*)

Request handler for the server status information.

Parameters **request** – aiohttp request object

Returns The web response

status_reset_handler (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e766535c0>*)

Request handler for resetting the timing statistics.

Parameters **request** – aiohttp request object

Returns The web response

stop () → None

Stop the webserver.

websocket_handler (*request*)

Send notifications to admin client over websocket.

class `aries_cloudagent.admin.server.AdminStatusSchema` (**args, **kwargs*)

Bases: `sphinx.ext.autodoc.importer._MockObject`

Schema for the status endpoint.

class `aries_cloudagent.admin.server.WebhookTarget` (*endpoint: str, topic_filter: Sequence[str] = None, max_attempts: int = None*)

Bases: `object`

Class for managing webhook target information.

topic_filter

Accessor for the target's topic filter.

1.1.2 aries_cloudagent.cache package

Submodules

aries_cloudagent.cache.base module

Abstract base classes for cache.

class aries_cloudagent.cache.base.BaseCache

Bases: `abc.ABC`

Abstract cache interface.

acquire (*key: str*)

Acquire a lock on a given cache key.

clear (*key: str*)

Remove an item from the cache, if present.

Parameters **key** – the key to remove

flush ()

Remove all items from the cache.

get (*key: str*)

Get an item from the cache.

Parameters **key** – the key to retrieve an item for

Returns The record found or *None*

release (*key: str*)

Release the lock on a given cache key.

set (*keys: Union[str, Sequence[str]], value: Any, ttl: int = None*)

Add an item to the cache with an optional ttl.

Parameters

- **keys** – the key or keys for which to set an item
- **value** – the value to store in the cache
- **ttl** – number of second that the record should persist

exception aries_cloudagent.cache.base.CacheError (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base class for cache-related errors.

class aries_cloudagent.cache.base.CacheKeyLock (cache: aries_cloudagent.cache.base.BaseCache, key: str)

Bases: `object`

A lock on a particular cache key.

Used to prevent multiple async threads from generating or querying the same semi-expensive data. Not thread safe.

done
Accessor for the done state.

future
Fetch the result in the form of an awaitable future.

parent
Accessor for the parent key lock, if any.

release()
Release the cache lock.

result
Fetch the current result, if any.

set_result (*value: Any, ttl: int = None*)
Set the result, updating the cache and any waiters.

aries_cloudagent.cache.basic module

Basic in-memory cache implementation.

class `aries_cloudagent.cache.basic.BasicCache`
Bases: `aries_cloudagent.cache.base.BaseCache`
Basic in-memory cache class.

clear (*key: str*)
Remove an item from the cache, if present.
Parameters **key** – the key to remove

flush ()
Remove all items from the cache.

get (*key: str*)
Get an item from the cache.
Parameters **key** – the key to retrieve an item for
Returns The record found or *None*

set (*keys: Union[str, Sequence[str]], value: Any, ttl: int = None*)
Add an item to the cache with an optional ttl.
Overwrites existing cache entries.
Parameters

- **keys** – the key or keys for which to set an item
- **value** – the value to store in the cache
- **ttl** – number of seconds that the record should persist

1.1.3 aries_cloudagent.commands package

Commands module common setup.

`aries_cloudagent.commands.available_commands()`
Index available commands.

`aries_cloudagent.commands.load_command(command: str)`

Load the module corresponding with a named command.

`aries_cloudagent.commands.run_command(command: str, argv: Sequence[str] = None)`

Execute a named command with command line arguments.

Submodules

`aries_cloudagent.commands.help` module

Help command for indexing available commands.

`aries_cloudagent.commands.help.execute(argv: Sequence[str] = None)`

Execute the help command.

`aries_cloudagent.commands.provision` module

Provision command for setting up agent settings before starting.

exception `aries_cloudagent.commands.provision.ProvisionError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Base exception for provisioning errors.

`aries_cloudagent.commands.provision.execute(argv: Sequence[str] = None)`

Entrypoint.

`aries_cloudagent.commands.provision.init_argument_parser(parser: parse.ArgumentParser)`

Initialize an argument parser with the module's arguments.

`aries_cloudagent.commands.provision.provision(settings: dict)`

Perform provisioning.

`aries_cloudagent.commands.start` module

Entrypoint.

`aries_cloudagent.commands.start.execute(argv: Sequence[str] = None)`

Entrypoint.

`aries_cloudagent.commands.start.init_argument_parser(parser: parse.ArgumentParser)`

Initialize an argument parser with the module's arguments.

`aries_cloudagent.commands.start.run_loop(startup: Coroutine[T_co, T_contra, V_co], shutdown: Coroutine[T_co, T_contra, V_co])`

Execute the application, handling signals and ctrl-c.

`aries_cloudagent.commands.start.shutdown_app(conductor: aries_cloudagent.core.conductor.Conductor)`

Shut down.

`aries_cloudagent.commands.start.start_app(conductor: aries_cloudagent.core.conductor.Conductor)`

Start up.

1.1.4 aries_cloudagent.config package

Submodules

aries_cloudagent.config.argparse module

Command line option parsing.

```
class aries_cloudagent.config.argparse.AdminGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup
    Admin server settings.

    CATEGORIES = ('start',)
    GROUP_NAME = 'Admin'

    add_arguments(parser: argparse.ArgumentParser)
        Add admin-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace)
        Extract admin settings.

class aries_cloudagent.config.argparse.ArgumentGroup
    Bases: abc.ABC
    A class representing a group of related command line arguments.

    GROUP_NAME = None

    add_arguments(parser: argparse.ArgumentParser)
        Add arguments to the provided argument parser.

    get_settings(args: argparse.Namespace) → dict
        Extract settings from the parsed arguments.

class aries_cloudagent.config.argparse.DebugGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup
    Debug settings.

    CATEGORIES = ('start',)
    GROUP_NAME = 'Debug'

    add_arguments(parser: argparse.ArgumentParser)
        Add debug command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract debug settings.

class aries_cloudagent.config.argparse.GeneralGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup
    General settings.

    CATEGORIES = ('general', 'start')
    GROUP_NAME = 'General'

    add_arguments(parser: argparse.ArgumentParser)
        Add general command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract general settings.
```

```
class aries_cloudagent.config.argparse.LedgerGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Ledger settings.

    CATEGORIES = ('start', 'general')

    GROUP_NAME = 'Ledger'

    add_arguments(parser: argparse.ArgumentParser)
        Add ledger-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract ledger settings.

class aries_cloudagent.config.argparse.LoggingGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Logging settings.

    CATEGORIES = ('general', 'start')

    GROUP_NAME = 'Logging'

    add_arguments(parser: argparse.ArgumentParser)
        Add logging-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Extract logging settings.

class aries_cloudagent.config.argparse.ProtocolGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Protocol settings.

    CATEGORIES = ('start',)

    GROUP_NAME = 'Protocol'

    add_arguments(parser: argparse.ArgumentParser)
        Add protocol-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace) → dict
        Get protocol settings.

class aries_cloudagent.config.argparse.TransportGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Transport settings.

    CATEGORIES = ('start',)

    GROUP_NAME = 'Transport'

    add_arguments(parser: argparse.ArgumentParser)
        Add transport-specific command line arguments to the parser.

    get_settings(args: argparse.Namespace)
        Extract transport settings.

class aries_cloudagent.config.argparse.WalletGroup
    Bases: aries_cloudagent.config.argparse.ArgumentGroup

    Wallet settings.

    CATEGORIES = ('general', 'start')
```

GROUP_NAME = 'Wallet'

add_arguments (*parser: argparse.ArgumentParser*)
Add wallet-specific command line arguments to the parser.

get_settings (*args: argparse.Namespace*) → dict
Extract wallet settings.

class aries_cloudagent.config.argparse.**group** (**categories*)
Bases: `object`

Decorator for registering argument groups.

classmethod **get_registered** (*category: str = None*)
Fetch the set of registered classes in a category.

aries_cloudagent.config.argparse.load_argument_groups (*parser: argparse.ArgumentParser, *groups*)

Log a set of argument groups into a parser.

Returns A callable to convert loaded arguments into a settings dictionary

aries_cloudagent.config.base module

Configuration base classes.

class aries_cloudagent.config.base.**BaseInjector**
Bases: `abc.ABC`

Base injector class.

copy () → aries_cloudagent.config.base.BaseInjector
Produce a copy of the injector instance.

inject (*base_cls: type, settings: Mapping[str, object] = None, *, required: bool = True*) → object
Get the provided instance of a given class identifier.

Parameters

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

Returns An instance of the base class, or None

class aries_cloudagent.config.base.**BaseProvider**
Bases: `abc.ABC`

Base provider class.

provide (*settings: aries_cloudagent.config.base.BaseSettings, injector: aries_cloudagent.config.base.BaseInjector*)
Provide the object instance given a config and injector.

class aries_cloudagent.config.base.**BaseSettings**
Bases: `collections.abc.Mapping`, `typing.Generic`

Base settings class.

copy () → aries_cloudagent.config.base.BaseSettings
Produce a copy of the settings instance.

extend (*other: Mapping[str, object]*) → aries_cloudagent.config.base.BaseSettings
Merge another mapping to produce a new settings instance.

get_bool (*var_names, default=None) → bool
Fetch a setting as a boolean value.

Parameters

- **var_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

get_int (*var_names, default=None) → int
Fetch a setting as an integer value.

Parameters

- **var_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

get_str (*var_names, default=None) → str
Fetch a setting as a string value.

Parameters

- **var_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

get_value (*var_names, default=None)
Fetch a setting.

Parameters

- **var_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

Returns The setting value, if defined, otherwise the default value

exception aries_cloudagent.config.base.**ConfigError** (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.core.error.BaseError*

A base exception for all configuration errors.

exception aries_cloudagent.config.base.**InjectorError** (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.config.base.ConfigError*

The base exception raised by *BaseInjector* implementations.

exception aries_cloudagent.config.base.**ProviderError** (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.config.base.ConfigError*

The base exception raised by *BaseProvider* implementations.

exception aries_cloudagent.config.base.**SettingsError** (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.config.base.ConfigError*

The base exception raised by *BaseSettings* implementations.

aries_cloudagent.config.base_context module

Base injection context builder classes.


```
class aries_cloudagent.config.base_context.ContextBuilder (settings: Mapping[str, object] = None)

    Bases: abc.ABC

    Base injection context builder class.

    build () → aries_cloudagent.config.injection_context.InjectionContext
        Build the new injection context.

    update_settings (settings: Mapping[str, object])
        Update the context builder with additional settings.
```

aries_cloudagent.config.default_context module

Classes for configuring the default injection context.

```
class aries_cloudagent.config.default_context.DefaultContextBuilder (settings: Mapping[str, object] = None)

    Bases: aries_cloudagent.config.base_context.ContextBuilder

    Default context builder.

    bind_providers (context: aries_cloudagent.config.injection_context.InjectionContext)
        Bind various class providers.

    build () → aries_cloudagent.config.injection_context.InjectionContext
        Build the new injection context.

    load_plugins (context: aries_cloudagent.config.injection_context.InjectionContext)
        Set up plugin registry and load plugins.
```

aries_cloudagent.config.error module

Errors for config modules.

```
exception aries_cloudagent.config.error.ArgsParseError (*args, error_code: str = None, **kwargs)

    Bases: aries_cloudagent.config.base.ConfigError

    Error raised when there is a problem parsing the command-line arguments.
```

aries_cloudagent.config.injection_context module

Injection context implementation.

```
class aries_cloudagent.config.injection_context.InjectionContext (*, settings: Mapping[str, object] = None, enforce_typing: bool = True)

    Bases: aries_cloudagent.config.base.BaseInjector

    Manager for configuration settings and class providers.

    ROOT_SCOPE = 'application'
```

copy() → aries_cloudagent.config.injection_context.InjectionContext
Produce a copy of the injector instance.

inject (*base_cls: type, settings: Mapping[str, object] = None, *, required: bool = True*) → object
Get the provided instance of a given class identifier.

Parameters

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

Returns An instance of the base class, or None

injector
Accessor for scope-specific injector.

injector_for_scope (*scope_name: str*) → aries_cloudagent.config.injector.Injector
Fetch the injector for a specific scope.

Parameters **scope_name** – The unique scope identifier

scope_name
Accessor for the current scope name.

settings
Accessor for scope-specific settings.

start_scope (*scope_name: str, settings: Mapping[str, object] = None*) → aries_cloudagent.config.injection_context.InjectionContext
Begin a new named scope.

Parameters

- **scope_name** – The unique name for the scope being entered
- **settings** – An optional mapping of additional settings to apply

Returns A new injection context representing the scope

update_settings (*settings: Mapping[str, object]*)
Update the scope with additional settings.

exception aries_cloudagent.config.injection_context.**InjectionContextError** (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.config.base.InjectorError*

Base class for issues in the injection context.

class aries_cloudagent.config.injection_context.**Scope** (*name, injector*)
Bases: *tuple*

injector
Alias for field number 1

name
Alias for field number 0

aries_cloudagent.config.injector module

Standard Injector implementation.

class aries_cloudagent.config.injector.**Injector** (*settings: Mapping[str, object] = None, enforce_typing: bool = True*)

Bases: *aries_cloudagent.config.base.BaseInjector*

Injector implementation with static and dynamic bindings.

bind_instance (*base_cls: type, instance: object*)

Add a static instance as a class binding.

bind_provider (*base_cls: type, provider: aries_cloudagent.config.base.BaseProvider, *, cache: bool = False*)

Add a dynamic instance resolver as a class binding.

clear_binding (*base_cls: type*)

Remove a previously-added binding.

copy () → aries_cloudagent.config.base.BaseInjector

Produce a copy of the injector instance.

get_provider (*base_cls: type*)

Find the provider associated with a class binding.

inject (*base_cls: type, settings: Mapping[str, object] = None, *, required: bool = True*)

Get the provided instance of a given class identifier.

Parameters

- **cls** – The base class to retrieve an instance of
- **params** – An optional dict providing configuration to the provider

Returns An instance of the base class, or None

settings

Accessor for scope-specific settings.

aries_cloudagent.config.ledger module

Ledger configuration.

aries_cloudagent.config.ledger.**accept_taa** (*ledger: aries_cloudagent.ledger.base.BaseLedger, taa_info, provision: bool = False*) → bool

Perform TAA acceptance.

aries_cloudagent.config.ledger.**fetch_genesis_transactions** (*genesis_url: str*) → str

Get genesis transactions.

aries_cloudagent.config.ledger.**ledger_config** (*context: aries_cloudagent.config.injection_context.InjectionContext, public_id: str, provision: bool = False*) → bool

Perform Indy ledger configuration.

aries_cloudagent.config.logging module

Utilities related to logging.

class `aries_cloudagent.config.logging.LoggingConfigurator`

Bases: `object`

Utility class used to configure logging and print an informative start banner.

classmethod `configure` (*logging_config_path: str = None, log_level: str = None, log_file: str = None*)

Configure logger.

Parameters

- **logging_config_path** – str: (Default value = None) Optional path to custom logging config
- **log_level** – str: (Default value = None)

classmethod `print_banner` (*agent_label, inbound_transports, outbound_transports, public_did, admin_server=None, banner_length=40, border_character=':'*)

Print a startup banner describing the configuration.

Parameters

- **agent_label** – Agent Label
- **inbound_transports** – Configured inbound transports
- **outbound_transports** – Configured outbound transports
- **admin_server** – Admin server info
- **public_did** – Public DID
- **banner_length** – (Default value = 40) Length of the banner
- **border_character** – (Default value = “:”) Character to use in banner
- **border** –

`aries_cloudagent.config.logging.load_resource` (*path: str, encoding: str = None*) → `TextIO`

Open a resource file located in a python package or the local filesystem.

Parameters **path** – The resource path in the form of *dir/file* or *package:dir/file*

Returns A file-like object representing the resource

aries_cloudagent.config.provider module

Service provider implementations.

class `aries_cloudagent.config.provider.CachedProvider` (*provider: aries_cloudagent.config.base.BaseProvider*)

Bases: `aries_cloudagent.config.base.BaseProvider`

Cache the result of another provider.

provide (*config: aries_cloudagent.config.base.BaseSettings, injector: aries_cloudagent.config.base.BaseInjector*)

Provide the object instance given a config and injector.

class `aries_cloudagent.config.provider.ClassProvider` (*instance_cls: Union[str, type], *ctor_args, async_init: str = None, **ctor_kwargs*)

Bases: `aries_cloudagent.config.base.BaseProvider`

Provider for a particular class.

```

class Inject (base_cls: type)
    Bases: object

    A class for passing injected arguments to the constructor.

    provide (config: aries_cloudagent.config.base.BaseSettings, injector:
               aries_cloudagent.config.base.BaseInjector)
        Provide the object instance given a config and injector.

class aries_cloudagent.config.provider.InstanceProvider (instance)
    Bases: aries_cloudagent.config.base.BaseProvider

    Provider for a previously-created instance.

    provide (config: aries_cloudagent.config.base.BaseSettings, injector:
               aries_cloudagent.config.base.BaseInjector)
        Provide the object instance given a config and injector.

class aries_cloudagent.config.provider.StatsProvider (provider:
                                                       aries_cloudagent.config.base.BaseProvider,
                                                       methods: Sequence[str], *, ig-
                                                       nore_missing: bool = True)
    Bases: aries_cloudagent.config.base.BaseProvider

    Add statistics to the results of another provider.

    provide (config: aries_cloudagent.config.base.BaseSettings, injector:
               aries_cloudagent.config.base.BaseInjector)
        Provide the object instance given a config and injector.

```

aries_cloudagent.config.settings module

Settings implementation.

```

class aries_cloudagent.config.settings.Settings (values: Mapping[str, object] = None)
    Bases: aries_cloudagent.config.base.BaseSettings

    Mutable settings implementation.

    clear_value (var_name: str)
        Remove a setting.

        Parameters var_name – The name of the setting

    copy () → aries_cloudagent.config.base.BaseSettings
        Produce a copy of the settings instance.

    extend (other: Mapping[str, object]) → aries_cloudagent.config.base.BaseSettings
        Merge another settings instance to produce a new instance.

    get_value (*var_names, default=None)
        Fetch a setting.

        Parameters
        • var_names – A list of variable name alternatives
        • default – The default value to return if none are defined

    set_default (var_name: str, value)
        Add a setting if not currently defined.

        Parameters

```

- **var_name** – The name of the setting
- **value** – The value to assign

set_value (*var_name: str, value*)

Add a setting.

Parameters

- **var_name** – The name of the setting
- **value** – The value to assign

aries_cloudagent.config.util module

Entrypoint.

class aries_cloudagent.config.util.**ByteSize** (*min_size: int = 0, max_size: int = 0*)

Bases: `object`

Argument value parser for byte sizes.

aries_cloudagent.config.util.**common_config** (*settings: Mapping[str, Any]*)

Perform common app configuration.

aries_cloudagent.config.wallet module

Wallet configuration.

aries_cloudagent.config.wallet.**wallet_config** (*context: aries_cloudagent.config.injection_context.InjectionContext, provision: bool = False*)

Initialize the wallet.

1.1.5 aries_cloudagent.core package

Submodules

aries_cloudagent.core.conductor module

The Conductor.

The conductor is responsible for coordinating messages that are received over the network, communicating with the ledger, passing messages to handlers, instantiating concrete implementations of required modules and storing data in the wallet.

class aries_cloudagent.core.conductor.**Conductor** (*context_builder: aries_cloudagent.config.base_context.ContextBuilder*)

Bases: `object`

Conductor class.

Class responsible for initializing concrete implementations of our require interfaces and routing inbound and outbound message data.

dispatch_complete (*message: aries_cloudagent.transport.inbound.message.InboundMessage, completed: aries_cloudagent.utils.task_queue.CompletedTask*)

Handle completion of message dispatch.

get_stats () → dict

Get the current stats tracked by the conductor.

handle_not_delivered (*context: aries_cloudagent.config.injection_context.InjectionContext, outbound: aries_cloudagent.transport.outbound.message.OutboundMessage*)
 Handle a message that failed delivery via outbound transports.

handle_not_returned (*context: aries_cloudagent.config.injection_context.InjectionContext, outbound: aries_cloudagent.transport.outbound.message.OutboundMessage*)
 Handle a message that failed delivery via an inbound session.

inbound_message_router (*message: aries_cloudagent.transport.inbound.message.InboundMessage, can_respond: bool = False*)
 Route inbound messages.

Parameters

- **message** – The inbound message instance
- **can_respond** – If the session supports return routing

outbound_message_router (*context: aries_cloudagent.config.injection_context.InjectionContext, outbound: aries_cloudagent.transport.outbound.message.OutboundMessage, inbound: aries_cloudagent.transport.inbound.message.InboundMessage = None*) → None
 Route an outbound message.

Parameters

- **context** – The request context
- **message** – An outbound message to be sent
- **inbound** – The inbound message that produced this response, if available

queue_outbound (*context: aries_cloudagent.config.injection_context.InjectionContext, outbound: aries_cloudagent.transport.outbound.message.OutboundMessage, inbound: aries_cloudagent.transport.inbound.message.InboundMessage = None*)
 Queue an outbound message.

Parameters

- **context** – The request context
- **message** – An outbound message to be sent
- **inbound** – The inbound message that produced this response, if available

setup ()
 Initialize the global request context.

start () → None
 Start the agent.

stop (*timeout=1.0*)
 Stop the agent.

webhook_router (*topic: str, payload: dict, endpoint: str, max_attempts: int = None*)
 Route a webhook through the outbound transport manager.

Parameters

- **topic** – The webhook topic
- **payload** – The webhook payload
- **endpoint** – The endpoint of the webhook target
- **max_attempts** – The maximum number of attempts

aries_cloudagent.core.dispatcher module

The Dispatcher.

The dispatcher is responsible for coordinating data flow between handlers, providing lifecycle hook callbacks storing state for message threads, etc.

class `aries_cloudagent.core.dispatcher.Dispatcher` (*context:*
aries_cloudagent.config.injection_context.InjectionContext)

Bases: `object`

Dispatcher class.

Class responsible for dispatching messages to message handlers and responding to other agents.

complete (*timeout: float = 0.1*)

Wait for pending tasks to complete.

handle_message (*inbound_message: aries_cloudagent.transport.inbound.message.InboundMessage,*
send_outbound: Coroutine[T_co, T_contra, V_co], send_webhook: Corou-
tine[T_co, T_contra, V_co] = None)

Configure responder and message context and invoke the message handler.

Parameters

- **inbound_message** – The inbound message instance
- **send_outbound** – Async function to send outbound messages
- **send_webhook** – Async function to dispatch a webhook

Returns The response from the handler

log_task (*task: aries_cloudagent.utils.task_queue.CompletedTask*)

Log a completed task using the stats collector.

make_message (*parsed_msg: dict*) → `aries_cloudagent.messaging.agent_message.AgentMessage`

Deserialize a message dict into the appropriate message instance.

Given a dict describing a message, this method returns an instance of the related message class.

Parameters **parsed_msg** – The parsed message

Returns An instance of the corresponding message class for this message

Raises

- `MessageParseError` – If the message doesn't specify @type
- `MessageParseError` – If there is no message class registered to handle
- the given type

put_task (*coro: Coroutine[T_co, T_contra, V_co], complete: Callable = None, ident: str = None*) →
`aries_cloudagent.utils.task_queue.PendingTask`

Run a task in the task queue, potentially blocking other handlers.

queue_message (*inbound_message: aries_cloudagent.transport.inbound.message.InboundMessage,*
send_outbound: Coroutine[T_co, T_contra, V_co], send_webhook: Corou-
tine[T_co, T_contra, V_co] = None, complete: Callable = None) →
`aries_cloudagent.utils.task_queue.PendingTask`

Add a message to the processing queue for handling.

Parameters

- **inbound_message** – The inbound message instance

- **send_outbound** – Async function to send outbound messages
- **send_webhook** – Async function to dispatch a webhook
- **complete** – Function to call when the handler has completed

Returns A pending task instance resolving to the handler task

run_task (*coro: Coroutine[T_co, T_contra, V_co]*, *complete: Callable = None*, *ident: str = None*) → *_asyncio.Task*
Run a task in the task queue, potentially blocking other handlers.

setup ()
Perform async instance setup.

class *aries_cloudagent.core.dispatcher.DispatcherResponder* (*context: aries_cloudagent.messaging.request_context.RequestContext*, *inbound_message: aries_cloudagent.transport.inbound.message.InboundMessage*, *send_outbound: Coroutine[T_co, T_contra, V_co]*, *send_webhook: Coroutine[T_co, T_contra, V_co] = None, **kwargs*)

Bases: *aries_cloudagent.messaging.responder.BaseResponder*

Handle outgoing messages from message handlers.

create_outbound (*message: Union[aries_cloudagent.messaging.agent_message.AgentMessage, str, bytes]*, ***kwargs*) → *aries_cloudagent.transport.outbound.message.OutboundMessage*
Create an OutboundMessage from a message body.

Parameters *message* – The message payload

send_outbound (*message: aries_cloudagent.transport.outbound.message.OutboundMessage*)
Send outbound message.

Parameters *message* – The *OutboundMessage* to be sent

send_webhook (*topic: str*, *payload: dict*)
Dispatch a webhook.

Parameters

- **topic** – the webhook topic identifier
- **payload** – the webhook payload value

aries_cloudagent.core.error module

Common exception classes.

exception *aries_cloudagent.core.error.BaseError* (**args*, *error_code: str = None*, ***kwargs*)

Bases: *Exception*

Generic exception class which other exceptions should inherit from.

error_code = **None**

message

Accessor for the error message.

exception `aries_cloudagent.core.error.StartupError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Error raised when there is a problem starting the conductor.

aries_cloudagent.core.plugin_registry module

Handle registration of plugin modules for extending functionality.

class `aries_cloudagent.core.plugin_registry.PluginRegistry`

Bases: `object`

Plugin registry for indexing application plugins.

init_context (context: `aries_cloudagent.config.injection_context.InjectionContext`)

Call plugin setup methods on the current context.

load_message_types (context: `aries_cloudagent.config.injection_context.InjectionContext`, plugin: `module`)

For modules that don't implement setup, register protocols manually.

plugin_names

Accessor for a list of all plugin modules.

plugins

Accessor for a list of all plugin modules.

register_admin_routes (app)

Call route registration methods on the current context.

register_package (package_name: str) → Sequence[module]

Register all modules (sub-packages) under a given package name.

register_plugin (module_name: str) → module

Register a plugin module.

aries_cloudagent.core.protocol_registry module

Handle registration and publication of supported protocols.

class `aries_cloudagent.core.protocol_registry.ProtocolRegistry`

Bases: `object`

Protocol registry for indexing message families.

controllers

Accessor for a list of all protocol controller functions.

message_types

Accessor for a list of all message types.

prepare_disclosed (context: `aries_cloudagent.config.injection_context.InjectionContext`, protocols: Sequence[str])

Call controllers and return publicly supported message families and roles.

protocols

Accessor for a list of all message protocols.

protocols_matching_query (*query: str*) → Sequence[str]

Return a list of message protocols matching a query string.

register_controllers (**controller_sets*)

Add new controllers.

Parameters *controller_sets* – Mappings of message families to coroutines

register_message_types (**typesets*)

Add new supported message types.

Parameters *typesets* – Mappings of message types to register

resolve_message_class (*message_type: str*) → type

Resolve a message_type to a message class.

Given a message type identifier, this method returns the corresponding registered message class.

Parameters *message_type* – Message type to resolve

Returns The resolved message class

1.1.6 aries_cloudagent.holder package

Submodules

aries_cloudagent.holder.base module

Base holder class.

class aries_cloudagent.holder.base.**BaseHolder**

Bases: `abc.ABC`

Base class for holder.

aries_cloudagent.holder.indy module

Indy issuer implementation.

class aries_cloudagent.holder.indy.**IndyHolder** (*wallet*)

Bases: `aries_cloudagent.holder.base.BaseHolder`

Indy holder class.

RECORD_TYPE_MIME_TYPES = 'attribute-mime-types'

create_credential_request (*credential_offer, credential_definition, did*)

Create a credential offer for the given credential definition id.

Parameters

- **credential_offer** – The credential offer to create request for
- **credential_definition** – The credential definition to create an offer for

Returns A credential request

create_presentation (*presentation_request: dict, requested_credentials: dict, schemas: dict, credential_definitions: dict*)

Get credentials stored in the wallet.

Parameters

- **presentation_request** – Valid indy format presentation request
- **requested_credentials** – Indy format requested_credentials
- **schemas** – Indy formatted schemas_json
- **credential_definitions** – Indy formatted schemas_json

delete_credential (*credential_id: str*)

Remove a credential stored in the wallet.

Parameters **credential_id** – Credential id to remove

get_credential (*credential_id: str*)

Get a credential stored in the wallet.

Parameters **credential_id** – Credential id to retrieve

get_credentials (*start: int, count: int, wql: dict*)

Get credentials stored in the wallet.

Parameters

- **start** – Starting index
- **count** – Number of records to return
- **wql** – wql query dict

get_credentials_for_presentation_request_by_referent (*presentation_request: dict, referents: Sequence[str], start: int, count: int, extra_query: dict = {}*)

Get credentials stored in the wallet.

Parameters

- **presentation_request** – Valid presentation request from issuer
- **referents** – Presentation request referents to use to search for creds
- **start** – Starting index
- **count** – Maximum number of records to return
- **extra_query** – wql query dict

get_mime_type (*credential_id: str, attr: str = None*) → Union[dict, str]

Get MIME type per attribute (or for all attributes).

Parameters

- **credential_id** – credential id
- **attr** – attribute of interest or omit for all

Returns: Attribute MIME type or dict mapping attribute names to MIME types `attr_meta_json = all_meta.tags.get(attr)`

store_credential (*credential_definition, credential_data, credential_request_metadata, credential_attr_mime_types=None, credential_id=None*)

Store a credential in the wallet.

Parameters

- **credential_definition** – Credential definition for this credential

- **credential_data** – Credential data generated by the issuer
- **credential_request_metadata** – credential request metadata generated by the issuer
- **credential_attr_mime_types** – dict mapping attribute names to (optional) MIME types to store as non-secret record, if specified

1.1.7 aries_cloudagent.issuer package

Submodules

aries_cloudagent.issuer.base module

Ledger issuer class.

class aries_cloudagent.issuer.base.**BaseIssuer**

Bases: `abc.ABC`

Base class for issuer.

aries_cloudagent.issuer.indy module

Indy issuer implementation.

class aries_cloudagent.issuer.indy.**IndyIssuer** (*wallet*)

Bases: `aries_cloudagent.issuer.base.BaseIssuer`

Indy issuer class.

create_credential (*schema, credential_offer, credential_request, credential_values*)

Create a credential.

Args *schema*: Schema to create credential for *credential_offer*: Credential Offer to create credential for *credential_request*: Credential request to create credential for *credential_values*: Values to go in credential

Returns A tuple of created credential, revocation id

create_credential_offer (*credential_definition_id: str*)

Create a credential offer for the given credential definition id.

Parameters *credential_definition_id* – The credential definition to create an offer for

Returns A credential offer

exception aries_cloudagent.issuer.indy.**IssuerError** (**args, error_code: str = None, **kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Generic issuer error.

aries_cloudagent.issuer.util module

1.1.8 aries_cloudagent.ledger package

Submodules

aries_cloudagent.ledger.base module

Ledger base class.

```
class aries_cloudagent.ledger.base.BaseLedger
```

Bases: `abc.ABC`

Base class for ledger.

LEDGER_TYPE = `None`

accept_txn_author_agreement (*taa_record: dict, mechanism: str, accept_time: int = None*)

Save a new record recording the acceptance of the TAA.

did_to_nym (*did: str*) → `str`

Remove the ledger's DID prefix to produce a nym.

fetch_txn_author_agreement ()

Fetch the current AML and TAA from the ledger.

get_endpoint_for_did (*did: str*) → `str`

Fetch the endpoint for a ledger DID.

Parameters *did* – The DID to look up on the ledger or in the cache

get_key_for_did (*did: str*) → `str`

Fetch the verkey for a ledger DID.

Parameters *did* – The DID to look up on the ledger or in the cache

get_latest_txn_author_acceptance ()

Look up the latest TAA acceptance.

get_txn_author_agreement (*reload: bool = False*)

Get the current transaction author agreement, fetching it if necessary.

nym_to_did (*nym: str*) → `str`

Format a nym with the ledger's DID prefix.

register_nym (*did: str, verkey: str, alias: str = None, role: str = None*)

Register a nym on the ledger.

Parameters

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

taa_digest (*version: str, text: str*)

Generate the digest of a TAA record.

update_endpoint_for_did (*did: str, endpoint: str*) → `bool`

Check and update the endpoint on the ledger.

Parameters

- **did** – The ledger DID
- **endpoint** – The endpoint address

aries_cloudagent.ledger.error module

Ledger related errors.

```
exception aries_cloudagent.ledger.error.BadLedgerRequestError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

The current request cannot proceed.

```
exception aries_cloudagent.ledger.error.ClosedPoolError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

Indy pool is closed.

```
exception aries_cloudagent.ledger.error.LedgerConfigError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

Base class for ledger configuration errors.

```
exception aries_cloudagent.ledger.error.LedgerError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base class for ledger errors.

```
exception aries_cloudagent.ledger.error.LedgerTransactionError(*args, error_code: str = None, **kwargs)
```

Bases: `aries_cloudagent.ledger.error.LedgerError`

The ledger rejected the transaction.

aries_cloudagent.ledger.indy module

Indy ledger implementation.

```
class aries_cloudagent.ledger.indy.IndyErrorHandler(message: str = None, error_cls: Type[aries_cloudagent.ledger.error.LedgerError] = <class 'aries_cloudagent.ledger.error.LedgerError'>)
```

Bases: `object`

Trap IndyError and raise an appropriate LedgerError instead.

```
classmethod wrap_error(err_value: <sphinx.ext.autodoc.importer.MockObject object at 0x7f9e75396eb8>, message: str = None, error_cls: Type[aries_cloudagent.ledger.error.LedgerError] = <class 'aries_cloudagent.ledger.error.LedgerError'>) → aries_cloudagent.ledger.error.LedgerError
```

Create an instance of LedgerError from an IndyError.

```
class aries_cloudagent.ledger.indy.IndyLedger(pool_name: str, wallet: aries_cloudagent.wallet.base.BaseWallet, *, keepalive: int = 0, cache: aries_cloudagent.cache.base.BaseCache = None, cache_duration: int = 600, read_only: bool = False)
```

Bases: `aries_cloudagent.ledger.base.BaseLedger`

Indy ledger class.

LEDGER_TYPE = 'indy'

accept_txn_author_agreement (*taa_record: dict, mechanism: str, accept_time: int = None*)

Save a new record recording the acceptance of the TAA.

check_existing_schema (*public_did: str, schema_name: str, schema_version: str, at-tribute_names: Sequence[str]*) → str

Check if a schema has already been published.

check_pool_config () → bool

Check if a pool config has been created.

close ()

Close the pool ledger.

create_pool_config (*genesis_transactions: str, recreate: bool = False*)

Create the pool ledger configuration.

credential_definition_id2schema_id (*credential_definition_id*)

From a credential definition, get the identifier for its schema.

Parameters **credential_definition_id** – The identifier of the credential definition from which to identify a schema

fetch_credential_definition (*credential_definition_id: str*)

Get a credential definition from the ledger by id.

Parameters **credential_definition_id** – The cred def id of the cred def to fetch

fetch_schema_by_id (*schema_id: str*)

Get schema from ledger.

Parameters **schema_id** – The schema id (or stringified sequence number) to retrieve

Returns Indy schema dict

fetch_schema_by_seq_no (*seq_no: int*)

Fetch a schema by its sequence number.

Parameters **seq_no** – schema ledger sequence number

Returns Indy schema dict

fetch_txn_author_agreement ()

Fetch the current AML and TAA from the ledger.

get_credential_definition (*credential_definition_id: str*)

Get a credential definition from the cache if available, otherwise the ledger.

Parameters **credential_definition_id** – The schema id of the schema to fetch cred def for

get_endpoint_for_did (*did: str*) → str

Fetch the endpoint for a ledger DID.

Parameters **did** – The DID to look up on the ledger or in the cache

get_indy_storage () → `aries_cloudagent.storage.indy.IndyStorage`

Get an IndyStorage instance for the current wallet.

get_key_for_did (*did: str*) → str

Fetch the verkey for a ledger DID.

Parameters **did** – The DID to look up on the ledger or in the cache

get_latest_txn_author_acceptance ()

Look up the latest TAA acceptance.

get_schema (*schema_id: str*)

Get a schema from the cache if available, otherwise fetch from the ledger.

Parameters **schema_id** – The schema id (or stringified sequence number) to retrieve

get_txn_author_agreement (*reload: bool = False*)

Get the current transaction author agreement, fetching it if necessary.

nym_to_did (*nym: str*) → str

Format a nym with the ledger's DID prefix.

open ()

Open the pool ledger, creating it if necessary.

register_nym (*did: str, verkey: str, alias: str = None, role: str = None*)

Register a nym on the ledger.

Parameters

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

send_credential_definition (*schema_id: str, tag: str = None*)

Send credential definition to ledger and store relevant key matter in wallet.

Parameters

- **schema_id** – The schema id of the schema to create cred def for
- **tag** – Option tag to distinguish multiple credential definitions

send_schema (*schema_name: str, schema_version: str, attribute_names: Sequence[str]*)

Send schema to ledger.

Parameters

- **schema_name** – The schema name
- **schema_version** – The schema version
- **attribute_names** – A list of schema attributes

taa_digest (*version: str, text: str*)

Generate the digest of a TAA record.

taa_rough_timestamp () → int

Get a timestamp accurate to the day.

Anything more accurate is a privacy concern.

update_endpoint_for_did (*did: str, endpoint: str*) → bool

Check and update the endpoint on the ledger.

Parameters

- **did** – The ledger DID
- **endpoint** – The endpoint address

- **transport_vk** – The endpoint transport verkey

aries_cloudagent.ledger.provider module

Default ledger provider classes.

```
class aries_cloudagent.ledger.provider.LedgerProvider
    Bases: aries_cloudagent.config.base.BaseProvider
    Provider for the default ledger implementation.

    LEDGER_CLASSES = {'indy': 'aries_cloudagent.ledger.indy.IndyLedger'}

    provide (settings: aries_cloudagent.config.base.BaseSettings, injector:
             aries_cloudagent.config.base.BaseInjector)
        Create and open the ledger instance.
```

aries_cloudagent.ledger.routes module

Ledger admin routes.

```
class aries_cloudagent.ledger.routes.AMLRecordSchema (*, only=None, ex-
                                                         clude=(), many=False, con-
                                                         text=None, load_only=(),
                                                         dump_only=(), partial=False,
                                                         unknown=None)

    Bases: marshmallow.schema.Schema
    Ledger AML record.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.ledger.routes.TAAAcceptSchema (*, only=None, ex-
                                                         clude=(), many=False, con-
                                                         text=None, load_only=(),
                                                         dump_only=(), partial=False,
                                                         unknown=None)

    Bases: marshmallow.schema.Schema
    Input schema for accepting the TAA.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.ledger.routes.TAAAcceptanceSchema (*, only=None, ex-
                                                         clude=(), many=False,
                                                         context=None,
                                                         load_only=(),
                                                         dump_only=(), partial=False,
                                                         un-
                                                         known=None)

    Bases: marshmallow.schema.Schema
    TAA acceptance record.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.ledger.routes.TAAInfoSchema (*, only=None, exclude=(),
                                                         many=False, context=None,
                                                         load_only=(), dump_only=(),
                                                         partial=False, unknown=None)

    Bases: marshmallow.schema.Schema
```

Transaction author agreement info.

opts = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.ledger.routes.TAARecordSchema(*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: marshmallow.schema.Schema

Ledger TAA record.

opts = <marshmallow.schema.SchemaOpts object>

```
class aries_cloudagent.ledger.routes.TAAResultSchema(*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: marshmallow.schema.Schema

Result schema for a transaction author agreement.

opts = <marshmallow.schema.SchemaOpts object>

```
aries_cloudagent.ledger.routes.get_did_endpoint(request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75b67d68>)
```

Request handler for getting a verkey for a DID from the ledger.

Parameters **request** – aiohttp request object

```
aries_cloudagent.ledger.routes.get_did_verkey(request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75b67d68>)
```

Request handler for getting a verkey for a DID from the ledger.

Parameters **request** – aiohttp request object

```
aries_cloudagent.ledger.routes.ledger_accept_taa(request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75b67d68>)
```

Request handler for accepting the current transaction author agreement.

Parameters **request** – aiohttp request object

Returns The DID list response

```
aries_cloudagent.ledger.routes.ledger_get_taa(request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75b67d68>)
```

Request handler for fetching the transaction author agreement.

Parameters **request** – aiohttp request object

Returns The TAA information including the AML

```
aries_cloudagent.ledger.routes.register(app: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75b67d68>)
```

Register routes.

```
aries_cloudagent.ledger.routes.register_ledger_nym(request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75b67d68>)
```

Request handler for registering a NYM with the ledger.

Parameters **request** – aiohttp request object

aries_cloudagent.ledger.util module

Ledger utilities.

1.1.9 aries_cloudagent.messaging package

Subpackages

aries_cloudagent.messaging.ack package

Submodules

aries_cloudagent.messaging.ack.message module

Represents an explicit ack message as per Aries RFC 15.

```
class aries_cloudagent.messaging.ack.message.Ack (status: str = None, **kwargs)
    Bases: aries_cloudagent.messaging.agent_message.AgentMessage
```

Base class representing an explicit ack message.

Subclass to adopt, specify Meta message type and handler class.

```
class Meta
```

Bases: *object*

Ack metadata.

```
schema_class = 'AckSchema'
```

```
class aries_cloudagent.messaging.ack.message.AckSchema (*args, **kwargs)
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
```

Schema for Ack base class.

```
class Meta
```

Bases: *object*

Ack schema metadata.

```
model_class
```

alias of *Ack*

```
status = <fields.Constant (default='OK', attribute=None, validate=None, required=True, ...)
```

aries_cloudagent.messaging.credential_definitions package

Submodules

aries_cloudagent.messaging.credential_definitions.routes module

Credential definition admin routes.

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionGetResu
```

Bases: `marshmallow.schema.Schema`

Results schema for schema get request.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSchema (
```

Bases: `marshmallow.schema.Schema`

Credential definition schema.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSendReq
```

Bases: `marshmallow.schema.Schema`

Request schema for schema send request.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSendRes
```

Bases: `marshmallow.schema.Schema`

Results schema for schema send request.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionsCreated
```

Bases: `marshmallow.schema.Schema`

Results schema for cred-defs-created request.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
aries_cloudagent.messaging.credential_definitions.routes.credential_definitions_created(req
```

Request handler for retrieving credential definitions that current agent created.

Parameters **request** – aiohttp request object

Returns The identifiers of matching credential definitions.

```
aries_cloudagent.messaging.credential_definitions.routes.credential_definitions_get_credent
```

Request handler for getting a credential definition from the ledger.

Parameters **request** – aiohttp request object

Returns The credential definition details.

`aries_cloudagent.messaging.credential_definitions.routes.credential_definitions_send_creden`

Request handler for sending a credential definition to the ledger.

Parameters `request` – aiohttp request object

Returns The credential definition identifier

`aries_cloudagent.messaging.credential_definitions.routes.register` (*app:*
<sphinx.ext.autodoc.importer._Mock
object at
0x7f9e7541d470>)

Register routes.

`aries_cloudagent.messaging.credential_definitions.util` module

Credential definition utilities.

`aries_cloudagent.messaging.decorators` package

Submodules

`aries_cloudagent.messaging.decorators.attach_decorator` module

A message decorator for attachments.

An attach decorator embeds content or specifies appended content.

```

class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator(*,
                                                                            ident:
                                                                              str
                                                                              =
                                                                              None,
                                                                              de-
                                                                              scrip-
                                                                              tion:
                                                                              str
                                                                              =
                                                                              None,
                                                                              file-
                                                                              name:
                                                                              str
                                                                              =
                                                                              None,
                                                                              mime_type:
                                                                              str
                                                                              =
                                                                              None,
                                                                              last-
                                                                              mod_time:
                                                                              str
                                                                              =
                                                                              None,
                                                                              byte_count:
                                                                              int
                                                                              =
                                                                              None,
                                                                              data:
                                                                              aries_cloudagent.m
                                                                              **kwargs)

```

Bases: `aries_cloudagent.messaging.models.base.BaseModel`

Class representing attach decorator.

```
class Meta
```

Bases: `object`

AttachDecorator metadata.

```
    schema_class = 'AttachDecoratorSchema'
```

```
classmethod from_indy_dict(indy_dict: dict, *, ident: str = None, description: str = None,
                           filename: str = None, lastmod_time: str = None, byte_count: int
                           = None)

```

Create *AttachDecorator* instance from indy object (dict).

Given indy object (dict), JSON dump, base64-encode, and embed it as data; mark *application/json* MIME type.

Parameters

- **indy_dict** – indy (dict) data structure
- **ident** – optional attachment identifier (default random UUID4)
- **description** – optional attachment description
- **filename** – optional attachment filename

- **lastmod_time** – optional attachment last modification time
- **byte_count** – optional attachment byte count

indy_dict

Return indy data structure encoded in attachment.

Returns: dict with indy object in data attachment

```
class aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorData(*,
                                                                              base64_:
                                                                              str
                                                                              =
                                                                              None,
                                                                              sig_:
                                                                              str
                                                                              =
                                                                              None,
                                                                              json_:
                                                                              str
                                                                              =
                                                                              None,
                                                                              links_:
                                                                              Union[list,
                                                                              str]
                                                                              =
                                                                              None,
                                                                              sha256_:
                                                                              str
                                                                              =
                                                                              None)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Attach decorator data.

class Meta

Bases: *object*

AttachDecoratorData metadata.

schema_class = 'AttachDecoratorDataSchema'

base64

Accessor for base64 decorator data, or None.

header (*idx: int = 0, jose: bool = True*) → Mapping[KT, VT_co]

Accessor for header info at input index, default 0 or unique for singly-signed.

Parameters

- **idx** – index of interest, zero-based (default 0)
- **jose** – True to return unprotected header attributes, False for protected only

json

Accessor for json decorator data, or None.

links

Accessor for links decorator data, or None.

sha256

Accessor for sha256 decorator data, or None.

sig

Accessor for signed-content decorator data, or None.

sign (*verkeys*: Union[str, Mapping[str, str]], *wallet*: aries_cloudagent.wallet.base.BaseWallet)

Sign and replace base64 data value of attachment.

Parameters

- **verkeys** – Verkey(s) of the signing party; specify: - single verkey alone for single signature with no key identifier (kid) - dict mapping single key identifier to verkey for single signature - dict mapping key identifiers to verkeys for multi-signature
- **wallet** – The wallet to use for the signature

signatures

Accessor for number of signatures.

signed

Accessor for signed content (payload), None for unsigned.

verify (*wallet*: aries_cloudagent.wallet.base.BaseWallet) → bool

Verify the signature(s).

Parameters **wallet** – Wallet to use to verify signature

Returns True if verification succeeds else False

class aries_cloudagent.messaging.decorators.attach_decorator.**AttachDecoratorDataSchema** (*args, **kwargs)

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Attach decorator data schema.

class Meta

Bases: *object*

Attach decorator data schema metadata.

model_class

alias of *AttachDecoratorData*

base64_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries

json_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re

links_ = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, re

sha256_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries

sig_ = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re

class aries_cloudagent.messaging.decorators.attach_decorator.**AttachDecoratorSchema** (*args, **kwargs)

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Attach decorator schema used in serialization/deserialization.

class Meta

Bases: *object*

AttachDecoratorSchema metadata.

model_class

alias of *AttachDecorator*

byte_count = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=N

```

data = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, re
description = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None
filename = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None
ident = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
lastmod_time = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
mime_type = <fields.String(default=<marshmallow.missing>, attribute=None, validate=Non

```

aries_cloudagent.messaging.decorators.base module

Classes for managing a collection of decorators.

```

class aries_cloudagent.messaging.decorators.base.BaseDecoratorSet (models:
                                                                    dict      =
                                                                    None)

    Bases: collections.OrderedDict

    Collection of decorators.

    add_model (key: str, model: Type[aries_cloudagent.messaging.models.base.BaseModel])
        Add a registered decorator model.

    copy () → aries_cloudagent.messaging.decorators.base.BaseDecoratorSet
        Return a copy of the decorator set.

    extract_decorators (message: Mapping[KT, VT_co], schema:
                        Type[<sphinx.ext.autodoc.importer._MockObject object at
                        0x7f9e760c42e8>] = None, serialized: bool = True, skip_attrs: Se-
                        quence[str] = None) → collections.OrderedDict
        Extract decorators and return the remaining properties.

    field (name: str) → aries_cloudagent.messaging.decorators.base.BaseDecoratorSet
        Access a named decorated field.

    fields
        Accessor for the set of currently defined fields.

    has_field (name: str) → bool
        Check for the existence of a named decorator field.

    load_decorator (key: str, value, serialized=False)
        Convert a decorator value to its loaded representation.

    models
        Accessor for the models dictionary.

    prefix
        Accessor for the decorator prefix.

    remove_field (name: str)
        Remove a named decorated field.

    remove_model (key: str)
        Remove a registered decorator model.

    to_dict (prefix: str = None) → collections.OrderedDict
        Convert to a dictionary (serialize).

    Raises BaseModelError – on decorator validation errors

```

```
exception aries_cloudagent.messaging.decorators.base.DecoratorError(*args,
                                                                    er-
                                                                    ror_code:
                                                                    str      =
                                                                    None,
                                                                    **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base error for decorator issues.

`aries_cloudagent.messaging.decorators.default` module

Default decorator set implementation.

```
class aries_cloudagent.messaging.decorators.default.DecoratorSet(models: dict
                                                                    = None)
```

Bases: `aries_cloudagent.messaging.decorators.base.BaseDecoratorSet`

Default decorator set implementation.

`aries_cloudagent.messaging.decorators.localization_decorator` module

The localization decorator (~110n) for message localization information.

```
class aries_cloudagent.messaging.decorators.localization_decorator.LocalizationDecorator(*,
```

Bases: `aries_cloudagent.messaging.models.base.BaseModel`

Class representing the localization decorator.

```
class Meta
```

Bases: `object`

LocalizationDecorator metadata.

```
schema_class = 'LocalizationDecoratorSchema'
```

class aries_cloudagent.messaging.decorators.localization_decorator.**LocalizationDecoratorSchema**

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Localization decorator schema used in serialization/deserialization.

class Meta

Bases: *object*

LocalizationDecoratorSchema metadata.

model_class

alias of *LocalizationDecorator*

catalogs

Used by autodoc_mock_imports.

locale

Used by autodoc_mock_imports.

localizable

Used by autodoc_mock_imports.

aries_cloudagent.messaging.decorators.please_ack_decorator module

The please-ack decorator to request acknowledgement.

class aries_cloudagent.messaging.decorators.please_ack_decorator.**PleaseAckDecorator** (*message_id*, *sequence_id*, *on:* *Sequence[str]*)

str

=

None,

on:

Se-

quence[str]

=

None)

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing the please-ack decorator.

class Meta

Bases: *object*

PleaseAckDecorator metadata.

schema_class = 'PleaseAckDecoratorSchema'

class aries_cloudagent.messaging.decorators.please_ack_decorator.**PleaseAckDecoratorSchema** (*message_id*, *sequence_id*, *on:* *Sequence[str]*)

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

PleaseAck decorator schema used in serialization/deserialization.

class Meta

Bases: *object*

PleaseAckDecoratorSchema metadata.

model_class

alias of *PleaseAckDecorator*

message_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)

```
on = <fields.List (default=<marshmallow.missing>, attribute=None, validate=None, require=True)
```

aries_cloudagent.messaging.decorators.signature_decorator module

Model and schema for working with field signatures within message bodies.

```
class aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator(*,
signature_type: str = None,
signature_data: str = None,
signer: str = None)
Bases: aries_cloudagent.messaging.models.base.BaseModel
```

Class representing a field value signed by a known verkey.

class Meta

Bases: `object`

SignatureDecorator metadata.

schema_class = 'SignatureDecoratorSchema'

TYPE_ED25519SHA512 = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/signature/1.0/ed25519Sha512_'

classmethod create (value, signer: str, wallet: aries_cloudagent.wallet.base.BaseWallet, timestamp=None) → aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator

Create a Signature.

Sign a field value and return a newly constructed *SignatureDecorator* representing the resulting signature.

Parameters

- **value** – Value to sign
- **signer** – Verkey of the signing party
- **wallet** – The wallet to use for the signature

Returns The created *SignatureDecorator* object

decode () -> (<class 'object'>, <class 'int'>)

Decode the signature to its timestamp and value.

Returns A tuple of (decoded message, timestamp)

verify (*wallet: aries_cloudagent.wallet.base.BaseWallet*) → bool
 Verify the signature against the signer's public key.

Parameters **wallet** – Wallet to use to verify signature

Returns True if verification succeeds else False

class `aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecoratorSchema` (*a

Bases: `aries_cloudagent.messaging.models.base.BaseModelSchema`

SignatureDecorator schema.

class **Meta**

Bases: `object`

SignatureDecoratorSchema metadata.

model_class

alias of `SignatureDecorator`

sig_data

Used by autodoc_mock_imports.

signature

Used by autodoc_mock_imports.

signature_type

Used by autodoc_mock_imports.

signer

Used by autodoc_mock_imports.

`aries_cloudagent.messaging.decorators.thread_decorator` module

A message decorator for threads.

A thread decorator identifies a message that may require additional context from previous messages.

```
class aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator (*,
                                                                              thid:
                                                                              str
                                                                              =
                                                                              None,
                                                                              pthid:
                                                                              str
                                                                              =
                                                                              None,
                                                                              sender_order:
                                                                              int
                                                                              =
                                                                              None,
                                                                              re-
                                                                              ceived_orders:
                                                                              Map-
                                                                              ping[KT,
                                                                              VT_co]
                                                                              =
                                                                              None)
```

Bases: `aries_cloudagent.messaging.models.base.BaseModel`

Class representing thread decorator.

class Meta

Bases: `object`

ThreadDecorator metadata.

schema_class = 'ThreadDecoratorSchema'

pthid

Accessor for parent thread identifier.

Returns This thread's *pthid*

received_orders

Get received orders.

Returns The highest sender_order value that the sender has seen from other sender(s) on the thread.

sender_order

Get sender order.

Returns A number that tells where this message fits in the sequence of all messages that the current sender has contributed to this thread

thid

Accessor for thread identifier.

Returns This thread's *thid*

class `aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecoratorSchema` (*args, **kwargs)

Bases: `aries_cloudagent.messaging.models.base.BaseModelSchema`

Thread decorator schema used in serialization/deserialization.

class Meta

Bases: `object`

ThreadDecoratorSchema metadata.

model_class

alias of `ThreadDecorator`

pthid

Used by autodoc_mock_imports.

received_orders

Used by autodoc_mock_imports.

sender_order

Used by autodoc_mock_imports.

thid

Used by autodoc_mock_imports.

`aries_cloudagent.messaging.decorators.timing_decorator` module

The timing decorator (~timing).

This decorator allows the timing of agent messages to be communicated and constrained.


```
class aries_cloudagent.messaging.decorators.timing_decorator.TimingDecorator(*,  
    in_time:  
        Union[str,  
            date-  
            time.datetime]  
    =  
    None,  
    out_time:  
        Union[str,  
            date-  
            time.datetime]  
    =  
    None,  
    stale_time:  
        Union[str,  
            date-  
            time.datetime]  
    =  
    None,  
    expires_time:  
        Union[str,  
            date-  
            time.datetime]  
    =  
    None,  
    delay_milli:  
        int  
    =  
    None,  
    wait_until_time:  
        Union[str,  
            date-  
            time.datetime]  
    =  
    None)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing the timing decorator.

```
class Meta  
    Bases: object  
    TimingDecorator metadata.  
    schema_class = 'TimingDecoratorSchema'
```

```
class aries_cloudagent.messaging.decorators.timing_decorator.TimingDecoratorSchema(*args,  
    **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Timing decorator schema used in serialization/deserialization.

```
class Meta  
    Bases: object  
    TimingDecoratorSchema metadata.
```

```

    model_class
        alias of TimingDecorator

delay_milli
    Used by autodoc_mock_imports.

expires_time
    Used by autodoc_mock_imports.

in_time
    Used by autodoc_mock_imports.

out_time
    Used by autodoc_mock_imports.

stale_time
    Used by autodoc_mock_imports.

wait_until_time
    Used by autodoc_mock_imports.

```

aries_cloudagent.messaging.decorators.transport_decorator module

The transport decorator (~transport).

This decorator allows changes to agent response behaviour and queue status updates.

```

class aries_cloudagent.messaging.decorators.transport_decorator.TransportDecorator (*,
                                                                 re-
                                                                 turn_route:
                                                                 str
                                                                 =
                                                                 None,
                                                                 re-
                                                                 turn_route:
                                                                 str
                                                                 =
                                                                 None,
                                                                 queued_me
                                                                 int
                                                                 =
                                                                 None)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing the transport decorator.

```

class Meta
    Bases: object

    TransportDecorator metadata.

    schema_class = 'TransportDecoratorSchema'

```

```

class aries_cloudagent.messaging.decorators.transport_decorator.TransportDecoratorSchema (*,
                                                                 **

```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Transport decorator schema used in serialization/deserialization.

```

class Meta
    Bases: object

```

TransportDecoratorSchema metadata.

model_class
alias of *TransportDecorator*

queued_message_count
Used by autodoc_mock_imports.

return_route
Used by autodoc_mock_imports.

return_route_thread
Used by autodoc_mock_imports.

aries_cloudagent.messaging.models package

Submodules

aries_cloudagent.messaging.models.base module

Base classes for Models and Schemas.

class aries_cloudagent.messaging.models.base.**BaseModel**
Bases: *abc.ABC*

Base model that provides convenience methods.

class **Meta**
Bases: *object*
BaseModel meta data.

schema_class = *None*

Schema
Accessor for the model's schema class.

Returns The schema class

classmethod **deserialize** (*obj*)
Convert from JSON representation to a model instance.

Parameters *obj* – The dict to load into a model instance

Returns A model instance for this data

classmethod **from_json** (*json_repr: Union[str, bytes]*)
Parse a JSON string into a model instance.

Parameters *json_repr* – JSON string

Returns A model instance representation of this JSON

serialize (*as_string=False*) → dict
Create a JSON-compatible dict representation of the model instance.

Parameters *as_string* – Return a string of JSON instead of a dict

Returns A dict representation of this model, or a JSON string if *as_string* is True

to_json () → str
Create a JSON representation of the model instance.

Returns A JSON representation of this message

exception `aries_cloudagent.messaging.models.base.BaseModelError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base exception class for base model errors.

class `aries_cloudagent.messaging.models.base.BaseModelSchema` (*args, **kwargs)
 Bases: `sphinx.ext.autodoc.importer._MockObject`

BaseModel schema.

class **Meta**
 Bases: `object`
 BaseModelSchema metadata.
model_class = None
ordered = True
skip_values = [None]

Model
 Accessor for the schema's model class.

Returns The model class

make_model (*data: dict*, **kwargs)
 Return model instance after loading.

Returns A model instance

remove_skipped_values (*data*, **kwargs)
 Remove values that are are marked to skip.

Returns Returns this modified data

skip_dump_only (*data*, **kwargs)
 Skip fields that are only expected during serialization.

Parameters **data** – The incoming data to clean

Returns The modified data

`aries_cloudagent.messaging.models.base.resolve_class` (*the_cls*, *relative_cls: type = None*)

Resolve a class.

Parameters

- **the_cls** – The class to resolve
- **relative_cls** – Relative class to resolve from

Returns The resolved class

Raises `ClassNotFoundError` – If the class could not be loaded

`aries_cloudagent.messaging.models.base.resolve_meta_property` (*obj*, *prop_name: str*, *defval=None*)

Resolve a meta property.

Parameters

- **prop_name** – The property to resolve
- **defval** – The default value

Returns The meta property

aries_cloudagent.messaging.models.base_record module

Classes for BaseStorage-based record management.

```
class aries_cloudagent.messaging.models.base_record.BaseRecord(id:      str =
                                                                None, state:
                                                                str = None,
                                                                *, created_at:
                                                                Union[str, date-
                                                                time.datetime]
                                                                = None, up-
                                                                dated_at:
                                                                Union[str, date-
                                                                time.datetime]
                                                                = None)
```

Bases: `aries_cloudagent.messaging.models.base.BaseModel`

Represents a single storage record.

CACHE_ENABLED = **False**

CACHE_TTL = **60**

LOG_STATE_FLAG = **None**

class Meta

Bases: `object`

BaseRecord metadata.

RECORD_ID_NAME = **'id'**

RECORD_TYPE = **None**

TAG_NAMES = **{'state'}**

WEBHOOK_TOPIC = **None**

classmethod cache_key (*record_id: str, record_type: str = None*)

Assemble a cache key.

Parameters

- **record_id** – The record identifier
- **The cache type identifier, defaulting to RECORD_TYPE** (*record_type*) –

clear_cached (*context: aries_cloudagent.config.injection_context.InjectionContext*)

Clear the cached value of this record, if any.

classmethod clear_cached_key (*context: aries_cloudagent.config.injection_context.InjectionContext,*
cache_key: str)

Shortcut method to clear a cached key value, if any.

Parameters

- **context** – The injection context to use

- **cache_key** – The unique cache identifier

delete_record (*context: aries_cloudagent.config.injection_context.InjectionContext*)
Remove the stored record.

Parameters context – The injection context to use

classmethod from_storage (*record_id: str, record: Mapping[str, Any]*)
Initialize a record from its stored representation.

Parameters

- **record_id** – The unique record identifier
- **record** – The stored representation

classmethod get_cached_key (*context: aries_cloudagent.config.injection_context.InjectionContext, cache_key: str*)
Shortcut method to fetch a cached key value.

Parameters

- **context** – The injection context to use
- **cache_key** – The unique cache identifier

classmethod get_tag_map () → Mapping[str, str]
Accessor for the set of defined tags.

classmethod log_state (*context: aries_cloudagent.config.injection_context.InjectionContext, msg: str, params: dict = None, override: bool = False*)
Print a message with increased visibility (for testing).

post_save (*context: aries_cloudagent.config.injection_context.InjectionContext, new_record: bool, last_state: str, webhook: bool = None*)
Perform post-save actions.

Parameters

- **context** – The injection context to use
- **new_record** – Flag indicating if the record was just created
- **last_state** – The previous state value
- **webhook** – Adjust whether the webhook is called

classmethod prefix_tag_filter (*tag_filter: dict*)
Prefix unencrypted tags used in the tag filter.

classmethod query (*context: aries_cloudagent.config.injection_context.InjectionContext, tag_filter: dict = None, post_filter: dict = None*) → Sequence[aries_cloudagent.messaging.models.base_record.BaseRecord]
Query stored records.

Parameters

- **context** – The injection context to use
- **tag_filter** – An optional dictionary of tag filter clauses
- **post_filter** – Additional value filters to apply

record_tags
Accessor to define implementation-specific tags.

record_value
Accessor to define custom properties for the JSON record value.

```
classmethod retrieve_by_id (context: aries_cloudagent.config.injection_context.InjectionContext,
                             record_id: str, cached: bool = True) →
                             aries_cloudagent.messaging.models.base_record.BaseRecord
```

Retrieve a stored record by ID.

Parameters

- **context** – The injection context to use
- **record_id** – The ID of the record to find
- **cached** – Whether to check the cache for this record

```
classmethod retrieve_by_tag_filter (context: aries_cloudagent.config.injection_context.InjectionContext,
                                     tag_filter: dict, post_filter: dict = None) →
                                     aries_cloudagent.messaging.models.base_record.BaseRecord
```

Retrieve a record by tag filter.

Parameters

- **context** – The injection context to use
- **tag_filter** – The filter dictionary to apply
- **post_filter** – Additional value filters to apply after retrieval

```
save (context: aries_cloudagent.config.injection_context.InjectionContext, *, reason: str = None,
       log_params: Mapping[str, Any] = None, log_override: bool = False, webhook: bool = None)
       → str
```

Persist the record to storage.

Parameters

- **context** – The injection context to use
- **reason** – A reason to add to the log
- **log_params** – Additional parameters to log
- **webhook** – Flag to override whether the webhook is sent

```
send_webhook (context: aries_cloudagent.config.injection_context.InjectionContext, payload: Any,
               topic: str = None)
```

Send a standard webhook.

Parameters

- **context** – The injection context to use
- **payload** – The webhook payload
- **topic** – The webhook topic, defaulting to WEBHOOK_TOPIC

```
classmethod set_cached_key (context: aries_cloudagent.config.injection_context.InjectionContext,
                             cache_key: str, value: Any, ttl=None)
```

Shortcut method to set a cached key value.

Parameters

- **context** – The injection context to use
- **cache_key** – The unique cache identifier
- **value** – The value to cache
- **ttl** – The cache ttl

storage_record

Accessor for a *StorageRecord* representing this record.

classmethod strip_tag_prefix (*tags: dict*)

Strip tilde from unencrypted tag names.

tags

Accessor for the record tags generated for this record.

value

Accessor for the JSON record value generated for this record.

webhook_payload

Return a JSON-serialized version of the record for the webhook.

webhook_topic

Return the webhook topic value.

```
class aries_cloudagent.messaging.models.base_record.BaseRecordSchema (*args,
                                                                    **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Schema to allow serialization/deserialization of base records.

class Meta

Bases: *object*

BaseRecordSchema metadata.

```
created_at = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<a
```

```
state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
```

```
updated_at = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<a
```

```
aries_cloudagent.messaging.models.base_record.match_post_filter(record: dict,
                                                                    post_filter:
                                                                    dict) → bool
```

Determine if a record value matches the post-filter.

aries_cloudagent.messaging.schemas package

Submodules

aries_cloudagent.messaging.schemas.routes module

Credential schema admin routes.


```
class aries_cloudagent.messaging.schemas.routes.SchemaGetResultsSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Results schema for schema get request.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.messaging.schemas.routes.SchemaSchema (*,
                                                                only=None,
                                                                exclude=(),
                                                                many=False,
                                                                context=None,
                                                                load_only=(),
                                                                dump_only=(),
                                                                partial=False,
                                                                unknown=None)

    Bases: marshmallow.schema.Schema

    Content for returned schema.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.messaging.schemas.routes.SchemaSendRequestSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Request schema for schema send request.

    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.messaging.schemas.routes.SchemaSendResultsSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Results schema for schema send request.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.messaging.schemas.routes.SchemasCreatedResultsSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Results schema for a schemas-created request.

opts = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.messaging.schemas.routes.register (app:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object at 0x7f9e75b56ef0>)
```

Register routes.

```
aries_cloudagent.messaging.schemas.routes.schemas_created (request:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object at
                                                                    0x7f9e75b56ef0>)
```

Request handler for retrieving schemas that current agent created.

Parameters **request** – aiohttp request object

Returns The identifiers of matching schemas

```
aries_cloudagent.messaging.schemas.routes.schemas_get_schema (request:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object at
                                                                    0x7f9e75b56ef0>)
```

Request handler for sending a credential offer.

Parameters **request** – aiohttp request object

Returns The schema details.

```
aries_cloudagent.messaging.schemas.routes.schemas_send_schema(request:
    <sphinx.ext.autodoc.importer._MockObject
    object at
    0x7f9e75b56ef0>)
```

Request handler for sending a credential offer.

Parameters **request** – aiohttp request object

Returns The schema id sent

aries_cloudagent.messaging.schemas.util module

Schema utilities.

Submodules

aries_cloudagent.messaging.agent_message module

Agent message base class and schema.

```
class aries_cloudagent.messaging.agent_message.AgentMessage(_id: str = None,
    _decorators:
    aries_cloudagent.messaging.decorators.base.
    = None)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Agent message base class.

Handler

Accessor for the agent message's handler class.

Returns Handler class

class Meta

Bases: *object*

AgentMessage metadata.

handler_class = None

message_type = None

schema_class = None

```
assign_thread_from(msg: aries_cloudagent.messaging.agent_message.AgentMessage)
```

Copy thread information from a previous message.

Parameters **msg** – The received message containing optional thread information

```
assign_thread_id(thid: str, pthid: str = None)
```

Assign a specific thread ID.

Parameters

- **thid** – The thread identifier
- **pthid** – The parent thread identifier

```
get_signature(field_name: str) → aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator
```

Get the signature for a named field.

Parameters `field_name` – Field name to get the signature for

Returns A SignatureDecorator for the requested field name

set_signature (*field_name: str, signature: aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator*
Add or replace the signature for a named field.

Parameters

- **field_name** – Field to set signature on
- **signature** – Signature for the field

sign_field (*field_name: str, signer_verkey: str, wallet: aries_cloudagent.wallet.base.BaseWallet, timestamp=None*) → *aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator*
Create and store a signature for a named field.

Parameters

- **field_name** – Field to sign
- **signer_verkey** – Verkey of signer
- **wallet** – Wallet to use for signature
- **timestamp** – Optional timestamp for signature

Returns A SignatureDecorator for newly created signature

Raises `ValueError` – If field_name doesn't exist on this message

verify_signatures (*wallet: aries_cloudagent.wallet.base.BaseWallet*) → *bool*
Verify all associated field signatures.

Parameters **wallet** – Wallet to use in verification

Returns True if all signatures verify, else false

verify_signed_field (*field_name: str, wallet: aries_cloudagent.wallet.base.BaseWallet, signer_verkey: str = None*) → *str*
Verify a specific field signature.

Parameters

- **field_name** – The field name to verify
- **wallet** – Wallet to use for the verification
- **signer_verkey** – Verkey of signer to use

Returns The verkey of the signer

Raises

- `ValueError` – If field_name does not exist on this message
- `ValueError` – If the verification fails
- `ValueError` – If the verkey of the signature does not match the provided verkey

exception `aries_cloudagent.messaging.agent_message.AgentMessageError` (**args, error_code: str = None, **kwargs*)

Bases: `aries_cloudagent.messaging.models.base.BaseModelError`

Base exception for agent message issues.

class `aries_cloudagent.messaging.agent_message.AgentMessageSchema` (**args*,
***kwargs*)

Bases: `aries_cloudagent.messaging.models.base.BaseModelSchema`

AgentMessage schema.

class `Meta`

Bases: `object`

AgentMessageSchema metadata.

model_class = `None`

signed_fields = `None`

check_dump_decorators (*obj*, ***kwargs*)

Pre-dump hook to validate and load the message decorators.

Parameters *obj* – The AgentMessage object

Raises `BaseModelError` – If a decorator does not validate

dump_decorators (*data*, ***kwargs*)

Post-dump hook to write the decorators to the serialized output.

Parameters *obj* – The serialized data

Returns The modified data

extract_decorators (*data*, ***kwargs*)

Pre-load hook to extract the decorators and check the signed fields.

Parameters *data* – Incoming data to parse

Returns Parsed and modified data

Raises

- `ValidationError` – If a field signature does not correlate
- to a field in the message
- `ValidationError` – If the message defines both a field signature
- and a value for the same field
- `ValidationError` – If there is a missing field signature

populate_decorators (*obj*, ***kwargs*)

Post-load hook to populate decorators on the message.

Parameters *obj* – The AgentMessage object

Returns The AgentMessage object with populated decorators

replace_signatures (*data*, ***kwargs*)

Post-dump hook to write the signatures to the serialized output.

Parameters *obj* – The serialized data

Returns The modified data

aries_cloudagent.messaging.base_handler module

A Base handler class for all message handlers.

class aries_cloudagent.messaging.base_handler.**BaseHandler**

Bases: `abc.ABC`

Abstract base class for handlers.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)

Abstract method for handler logic.

Parameters

- **context** – Request context object
- **responder** – A responder object

exception aries_cloudagent.messaging.base_handler.**HandlerException** (**args*, *error_code:* `str` = `None`, ***kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Exception base class for generic handler errors.

aries_cloudagent.messaging.error module

Messaging-related error classes and codes.

exception aries_cloudagent.messaging.error.**MessageParseError** (**args*, *error_code:* `str` = `None`, ***kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Message parse error.

error_code = 'message_parse_error'

exception aries_cloudagent.messaging.error.**MessagePrepareError** (**args*, *error_code:* `str` = `None`, ***kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Message preparation error.

error_code = 'message_prepare_error'

aries_cloudagent.messaging.request_context module

Request context class.

A request context provides everything required by handlers and other parts of the system to process a message.

```
class aries_cloudagent.messaging.request_context.RequestContext (*,
                                                                base_context:
                                                                aries_cloudagent.config.injection_context.InjectionContext
                                                                = None, settings: Mapping[str, object] =
                                                                None)
```

Bases: `aries_cloudagent.config.injection_context.InjectionContext`

Context established by the Conductor and passed into message handlers.

connection_ready

Accessor for the flag indicating an active connection with the sender.

Returns True if the connection is active, else False

connection_record

Accessor for the related connection record.

copy () → `aries_cloudagent.messaging.request_context.RequestContext`

Produce a copy of the request context instance.

default_endpoint

Accessor for the default agent endpoint (from agent config).

Returns The default agent endpoint

default_label

Accessor for the default agent label (from agent config).

Returns The default label

message

Accessor for the deserialized message instance.

Returns This context's agent message

message_receipt

Accessor for the message receipt information.

Returns This context's message receipt information

aries_cloudagent.messaging.responder module

A message responder.

The responder is provided to message handlers to enable them to send a new message in response to the message being handled.

```
class aries_cloudagent.messaging.responder.BaseResponder (*, connection_id: str =
                                                                None, reply_session_id:
                                                                str = None, reply_to_verkey:
                                                                str = None)
```

Bases: `abc.ABC`

Interface for message handlers to send responses.

create_outbound (*message: Union[aries_cloudagent.messaging.agent_message.AgentMessage, str, bytes], *, connection_id: str = None, reply_session_id: str = None, reply_thread_id: str = None, reply_to_verkey: str = None, target: aries_cloudagent.connections.models.connection_target.ConnectionTarget = None, target_list: Sequence[aries_cloudagent.connections.models.connection_target.ConnectionTarget] = None, to_session_only: bool = False*) → *aries_cloudagent.transport.outbound.message.OutboundMessage*
Create an OutboundMessage from a message payload.

send (*message: Union[aries_cloudagent.messaging.agent_message.AgentMessage, str, bytes], **kwargs*)
Convert a message to an OutboundMessage and send it.

send_outbound (*message: aries_cloudagent.transport.outbound.message.OutboundMessage*)
Send an outbound message.

Parameters *message* – The *OutboundMessage* to be sent

send_reply (*message: Union[aries_cloudagent.messaging.agent_message.AgentMessage, str, bytes], *, connection_id: str = None, target: aries_cloudagent.connections.models.connection_target.ConnectionTarget = None, target_list: Sequence[aries_cloudagent.connections.models.connection_target.ConnectionTarget] = None*)
Send a reply to an incoming message.

Parameters

- **message** – the *AgentMessage*, or pre-packed str or bytes to reply with
- **connection_id** – optionally override the target connection ID
- **target** – optionally specify a *ConnectionTarget* to send to

Raises *ResponderError* – If there is no active connection

send_webhook (*topic: str, payload: dict*)
Dispatch a webhook.

Parameters

- **topic** – the webhook topic identifier
- **payload** – the webhook payload value

class *aries_cloudagent.messaging.responder.MockResponder*
Bases: *aries_cloudagent.messaging.responder.BaseResponder*
Mock responder implementation for use by tests.

send (*message: Union[aries_cloudagent.messaging.agent_message.AgentMessage, str, bytes], **kwargs*)
Convert a message to an OutboundMessage and send it.

send_outbound (*message: aries_cloudagent.transport.outbound.message.OutboundMessage*)
Send an outbound message.

send_reply (*message: Union[aries_cloudagent.messaging.agent_message.AgentMessage, str, bytes], **kwargs*)
Send a reply to an incoming message.

send_webhook (*topic: str, payload: dict*)
Send an outbound message.

exception `aries_cloudagent.messaging.responder.ResponderError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Responder error.

`aries_cloudagent.messaging.util` module

Utils for messages.

`aries_cloudagent.messaging.util.canon` (*raw_attr_name*: str) → str
Canonicalize input attribute name for indy proofs and credential offers.

Parameters *raw_attr_name* – raw attribute name

Returns canonicalized attribute name

`aries_cloudagent.messaging.util.datetime_now`() → `datetime.datetime`
Timestamp in UTC.

`aries_cloudagent.messaging.util.datetime_to_str` (*dt*: Union[str, `datetime.datetime`]) → str
Convert a datetime object to an indy-standard datetime string.

Parameters *dt* – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.encode` (*orig*: Any) → str
Encode a credential value as an int.

Encode credential attribute value, purely stringifying any int32 and leaving numeric int32 strings alone, but mapping any other input to a stringified 256-bit (but not 32-bit) integer. Predicates in indy-sdk operate on int32 values properly only when their encoded values match their raw values.

Parameters *orig* – original value to encode

Returns encoded value

`aries_cloudagent.messaging.util.epoch_to_str` (*epoch*: int) → str
Convert epoch seconds to indy-standard datetime string.

Parameters *epoch* – epoch seconds

`aries_cloudagent.messaging.util.str_to_datetime` (*dt*: Union[str, `datetime.datetime`]) → `datetime.datetime`
Convert an indy-standard datetime string to a datetime.

Using a fairly lax regex pattern to match slightly different formats. In Python 3.7 `datetime.fromisoformat` might be used.

Parameters *dt* – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.str_to_epoch` (*dt*: Union[str, `datetime.datetime`]) → int
Convert an indy-standard datetime string to epoch seconds.

Parameters *dt* – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.time_now`() → str
Timestamp in ISO format.

aries_cloudagent.messaging.valid module

Validators for schema fields.

```
class aries_cloudagent.messaging.valid.Base64
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate base64 value.

```
EXAMPLE = 'ey4uLn0='
```

```
class aries_cloudagent.messaging.valid.Base64URL
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate base64 value.

```
EXAMPLE = 'ey4uLn0='
```

```
class aries_cloudagent.messaging.valid.IndyCredDefId
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against indy credential definition identifier specification.

```
EXAMPLE = 'WgWxqztrNooG92RXvxSTWv:3:CL:20:tag'
```

```
class aries_cloudagent.messaging.valid.IndyDID
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against indy DID.

```
EXAMPLE = 'WgWxqztrNooG92RXvxSTWv'
```

```
class aries_cloudagent.messaging.valid.IndyISO8601DateTime
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against ISO 8601 datetime format, indy profile.

```
EXAMPLE = '2020-03-09 21:47:11Z'
```

```
class aries_cloudagent.messaging.valid.IndyPredicate
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against indy predicate.

```
EXAMPLE = '>='
```

```
class aries_cloudagent.messaging.valid.IndyRawPublicKey
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against indy (Ed25519VerificationKey2018) raw public key.

```
EXAMPLE = 'H3C2AVvLMv6gmMnam3uVAjZpfkcJCwDwnZn6z3wXmqPV'
```

```
class aries_cloudagent.messaging.valid.IndyRevRegId
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against indy revocation registry identifier specification.

```
EXAMPLE = 'WgWxqztrNooG92RXvxSTWv:4:WgWxqztrNooG92RXvxSTWv:3:CL:20:tag:CL_ACCUM:0'
```

```
class aries_cloudagent.messaging.valid.IndySchemaId
    Bases: sphinx.ext.autodoc.importer._MockObject
```

Validate value against indy schema identifier specification.

```
EXAMPLE = 'WgWxqztrNooG92RXvxSTWv:2:schema_name:1.0'
```

```

class aries_cloudagent.messaging.valid.IndyVersion
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate value against indy version specification.
    EXAMPLE = '1.0'

class aries_cloudagent.messaging.valid.IntEpoch
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate value against (integer) epoch format.
    EXAMPLE = 1583790431

class aries_cloudagent.messaging.valid.SHA256Hash
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate (binhex-encoded) SHA256 value.
    EXAMPLE = '617a48c7c8afe0521efdc03e5bb0ad9e655893e6b4b51f0e794d70fba132aacb'

class aries_cloudagent.messaging.valid.UUIDFour
    Bases: sphinx.ext.autodoc.importer._MockObject
    Validate UUID4: 8-4-4-4-12 hex digits, the 13th of which being 4.
    EXAMPLE = '3fa85f64-5717-4562-b3fc-2c963f66afa6'

```

1.1.10 aries_cloudagent.storage package

Submodules

aries_cloudagent.storage.base module

Abstract base classes for non-secrets storage.

```

class aries_cloudagent.storage.base.BaseStorage
    Bases: abc.ABC
    Abstract Non-Secrets interface.

    add_record(record: aries_cloudagent.storage.record.StorageRecord)
        Add a new record to the store.

        Parameters record – StorageRecord to be stored

    delete_record(record: aries_cloudagent.storage.record.StorageRecord)
        Delete an existing record.

        Parameters record – StorageRecord to delete

    delete_record_tags(record: aries_cloudagent.storage.record.StorageRecord, tags: typing.Sequence, typing.Mapping)
        Update an existing stored record's tags.

        Parameters
        • record – StorageRecord to delete
        • tags – Tags

    get_record(record_type: str, record_id: str, options: Mapping[KT, VT_co] = None) →
        aries_cloudagent.storage.record.StorageRecord
        Fetch a record from the store by type and ID.

```

Parameters

- **record_type** – The record type
- **record_id** – The record id
- **options** – A dictionary of backend-specific options

Returns A *StorageRecord* instance

search_records (*type_filter*: str, *tag_query*: Mapping[KT, VT_co] = None, *page_size*: int = None, *options*: Mapping[KT, VT_co] = None) → aries_cloudagent.storage.base.BaseStorageRecordSearch
Create a new record query.

Parameters

- **type_filter** – Filter string
- **tag_query** – Tags to query
- **page_size** – Page size
- **options** – Dictionary of backend-specific options

Returns An instance of *BaseStorageRecordSearch*

update_record_tags (*record*: aries_cloudagent.storage.record.StorageRecord, *tags*: Mapping[KT, VT_co])
Update an existing stored record's tags.

Parameters

- **record** – *StorageRecord* to update
- **tags** – New tags

update_record_value (*record*: aries_cloudagent.storage.record.StorageRecord, *value*: str)
Update an existing stored record's value.

Parameters

- **record** – *StorageRecord* to update
- **value** – The new value

```
class aries_cloudagent.storage.base.BaseStorageRecordSearch (store:
    aries_cloudagent.storage.base.BaseStorage,
    type_filter: str,
    tag_query: Mapping[KT, VT_co],
    page_size: int = None, options: Mapping[KT, VT_co] = None)
```

Bases: `abc.ABC`

Represent an active stored records search.

close()

Dispose of the search query.

fetch (*max_count*: int) → Sequence[aries_cloudagent.storage.record.StorageRecord]

Fetch the next list of results from the store.

Parameters **max_count** – Max number of records to return

Returns A list of *StorageRecord*

fetch_all () → Sequence[aries_cloudagent.storage.record.StorageRecord]
Fetch all records from the query.

fetch_single () → aries_cloudagent.storage.record.StorageRecord
Fetch a single query result.

handle
Handle a search request.

open ()
Start the search query.

opened
Accessor for open state.

Returns True if opened, else False

option (name: str, default=None)
Fetch a named search option, if defined.

Returns The option value or default

options
Accessor for the search options.

Returns The search options

page_size
Accessor for page size.

Returns The page size

store
BaseStorage backend for this implementation.

Returns The *BaseStorage* implementation being used

tag_query
Accessor for tag query.

Returns The tag query

type_filter
Accessor for type filter.

Returns The type filter

aries_cloudagent.storage.basic module

Basic in-memory storage implementation (non-wallet).

```
class aries_cloudagent.storage.basic.BasicStorage (_wallet:
                                                    aries_cloudagent.wallet.base.BaseWallet
                                                    = None)
```

Bases: *aries_cloudagent.storage.base.BaseStorage*

Basic in-memory storage class.

add_record (record: aries_cloudagent.storage.record.StorageRecord)
Add a new record to the store.

Parameters **record** – *StorageRecord* to be stored

Raises

- `StorageError` – If no record is provided
- `StorageError` – If the record has no ID

delete_record (*record: aries_cloudagent.storage.record.StorageRecord*)

Delete a record.

Parameters **record** – *StorageRecord* to delete

Raises `StorageNotFoundError` – If record not found

delete_record_tags (*record: aries_cloudagent.storage.record.StorageRecord, tags: (typing.Sequence, typing.Mapping)*)

Update an existing stored record's tags.

Parameters

- **record** – *StorageRecord* to delete
- **tags** – Tags

Raises `StorageNotFoundError` – If record not found

get_record (*record_type: str, record_id: str, options: Mapping[KT, VT_co] = None*) → *aries_cloudagent.storage.record.StorageRecord*

Fetch a record from the store by type and ID.

Parameters

- **record_type** – The record type
- **record_id** – The record id
- **options** – A dictionary of backend-specific options

Returns A *StorageRecord* instance

Raises `StorageNotFoundError` – If the record is not found

search_records (*type_filter: str, tag_query: Mapping[KT, VT_co] = None, page_size: int = None, options: Mapping[KT, VT_co] = None*) → *aries_cloudagent.storage.basic.BasicStorageRecordSearch*

Search stored records.

Parameters

- **type_filter** – Filter string
- **tag_query** – Tags to query
- **page_size** – Page size
- **options** – Dictionary of backend-specific options

Returns An instance of *BaseStorageRecordSearch*

update_record_tags (*record: aries_cloudagent.storage.record.StorageRecord, tags: Mapping[KT, VT_co]*)

Update an existing stored record's tags.

Parameters

- **record** – *StorageRecord* to update
- **tags** – New tags

Raises `StorageNotFoundError` – If record not found

update_record_value (*record: aries_cloudagent.storage.record.StorageRecord, value: str*)
Update an existing stored record's value.

Parameters

- **record** – *StorageRecord* to update
- **value** – The new value

Raises *StorageNotFoundError* – If record not found

```
class aries_cloudagent.storage.basic.BasicStorageRecordSearch (store:
                                                    aries_cloudagent.storage.basic.BasicStorageRecordSearch,
                                                    type_filter: str,
                                                    tag_query: Mapping[KT, VT_co],
                                                    page_size: int =
                                                    None, options:
                                                    Mapping[KT,
                                                    VT_co] = None)
```

Bases: *aries_cloudagent.storage.base.BaseStorageRecordSearch*

Represent an active stored records search.

close ()
Dispose of the search query.

fetch (*max_count: int*) → *Sequence[aries_cloudagent.storage.record.StorageRecord]*
Fetch the next list of results from the store.

Parameters **max_count** – Max number of records to return

Returns A list of *StorageRecord*

Raises *StorageSearchError* – If the search query has not been opened

open ()
Start the search query.

opened
Accessor for open state.

Returns True if opened, else False

```
aries_cloudagent.storage.basic.basic_tag_query_match (tags: dict, tag_query: dict) →
                                                         bool
```

Match simple tag filters (string values).

```
aries_cloudagent.storage.basic.basic_tag_value_match (value: str, match: dict) → bool  
Match a single tag against a tag subquery.
```

TODO: What type coercion is needed? (support int or float values?)

aries_cloudagent.storage.error module

Storage-related exceptions.

```
exception aries_cloudagent.storage.error.StorageDuplicateError (*args, error_code:
                                                                str = None,
                                                                **kwargs)
```

Bases: *aries_cloudagent.storage.error.StorageError*

Duplicate record found in storage.

exception `aries_cloudagent.storage.error.StorageError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Base class for Storage errors.

exception `aries_cloudagent.storage.error.StorageNotFoundError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.storage.error.StorageError`

Record not found in storage.

exception `aries_cloudagent.storage.error.StorageSearchError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.storage.error.StorageError`

General exception during record search.

aries_cloudagent.storage.indy module

Indy implementation of BaseStorage interface.

class `aries_cloudagent.storage.indy.IndyStorage(wallet: aries_cloudagent.wallet.indy.IndyWallet)`

Bases: `aries_cloudagent.storage.base.BaseStorage`

Indy Non-Secrets interface.

add_record (`record: aries_cloudagent.storage.record.StorageRecord`)
Add a new record to the store.

Parameters **record** – *StorageRecord* to be stored

delete_record (`record: aries_cloudagent.storage.record.StorageRecord`)
Delete a record.

Parameters **record** – *StorageRecord* to delete

Raises

- *StorageNotFoundError* – If record not found
- *StorageError* – If a libindy error occurs

delete_record_tags (`record: aries_cloudagent.storage.record.StorageRecord, tags: typing.Sequence, typing.Mapping`)
Update an existing stored record's tags.

Parameters

- **record** – *StorageRecord* to delete
- **tags** – Tags

get_record (`record_type: str, record_id: str, options: Mapping[KT, VT_co] = None`) → `aries_cloudagent.storage.record.StorageRecord`
Fetch a record from the store by type and ID.

Parameters

- **record_type** – The record type
- **record_id** – The record id

- **options** – A dictionary of backend-specific options

Returns A *StorageRecord* instance

Raises

- *StorageError* – If the record is not provided
- *StorageError* – If the record ID not provided
- *StorageNotFoundError* – If the record is not found
- *StorageError* – If record not found

search_records (*type_filter*: *str*, *tag_query*: *Mapping[KT, VT_co]* = *None*,
page_size: *int* = *None*, *options*: *Mapping[KT, VT_co]* = *None*) →
 aries_cloudagent.storage.indy.IndyStorageRecordSearch
 Search stored records.

Parameters

- **type_filter** – Filter string
- **tag_query** – Tags to query
- **page_size** – Page size
- **options** – Dictionary of backend-specific options

Returns An instance of *IndyStorageRecordSearch*

update_record_tags (*record*: *aries_cloudagent.storage.record.StorageRecord*, *tags*: *Mapping[KT, VT_co]*)
 Update an existing stored record's tags.

Parameters

- **record** – *StorageRecord* to update
- **tags** – New tags

Raises

- *StorageNotFoundError* – If record not found
- *StorageError* – If a libindy error occurs

update_record_value (*record*: *aries_cloudagent.storage.record.StorageRecord*, *value*: *str*)
 Update an existing stored record's value.

Parameters

- **record** – *StorageRecord* to update
- **value** – The new value

Raises

- *StorageNotFoundError* – If record not found
- *StorageError* – If a libindy error occurs

wallet

Accessor for *IndyWallet* instance.

```
class aries_cloudagent.storage.indy.IndyStorageRecordSearch (store:
    aries_cloudagent.storage.indy.IndyStorage,
    type_filter: str,
    tag_query: Mapping[KT, VT_co],
    page_size: int =
    None, options: Mapping[KT, VT_co] =
    None)
```

Bases: `aries_cloudagent.storage.base.BaseStorageRecordSearch`

Represent an active stored records search.

close()
Dispose of the search query.

fetch (*max_count: int*) → Sequence[`aries_cloudagent.storage.record.StorageRecord`]
Fetch the next list of results from the store.

Parameters **max_count** – Max number of records to return

Returns A list of *StorageRecord*

Raises `StorageSearchError` – If the search query has not been opened

handle
Accessor for search handle.

Returns The handle

open()
Start the search query.

opened
Accessor for open state.

Returns True if opened, else False

aries_cloudagent.storage.provider module

Default storage provider classes.

```
class aries_cloudagent.storage.provider.StorageProvider
    Bases: aries_cloudagent.config.base.BaseProvider
    Provider for the default configurable storage classes.
    STORAGE_TYPES = {'basic': 'aries_cloudagent.storage.basic.BasicStorage', 'indy': 'ar
    provide (settings: aries_cloudagent.config.base.BaseSettings, injector:
        aries_cloudagent.config.base.BaseInjector)
        Create and return the storage instance.
```

aries_cloudagent.storage.record module

Record instance stored and searchable by BaseStorage implementation.

```
class aries_cloudagent.storage.record.StorageRecord
    Bases: aries_cloudagent.storage.record.StorageRecord
    Storage record class.
```

1.1.11 aries_cloudagent.transport package

Subpackages

aries_cloudagent.transport.inbound package

Submodules

aries_cloudagent.transport.inbound.base module

Base inbound transport class.

```
class aries_cloudagent.transport.inbound.base.BaseInboundTransport (scheme:
                                                                    str, create_session:
                                                                    Callable,
                                                                    *,
                                                                    max_message_size:
                                                                    int = 0,
                                                                    wire_format:
                                                                    aries_cloudagent.transport.wire_format.BaseWireFormat = None)
```

Bases: `abc.ABC`

Base inbound transport class.

create_session (*, accept_undelivered: bool = False, can_respond: bool = False, client_info: dict = None, wire_format: aries_cloudagent.transport.wire_format.BaseWireFormat = None) → Awaitable[aries_cloudagent.transport.inbound.session.InboundSession]
Create a new inbound session.

Parameters

- **accept_undelivered** – Flag for accepting undelivered messages
- **can_respond** – Flag indicating that the transport can send responses
- **client_info** – Request-specific client information
- **wire_format** – Optionally override the session wire format

max_message_size

Accessor for this transport's max message size.

scheme

Accessor for this transport's scheme.

start () → None

Start listening for on this transport.

stop () → None

Stop listening for on this transport.

```
class aries_cloudagent.transport.inbound.base.InboundTransportConfiguration (module,
                                                                    host,
                                                                    port)
```

Bases: `tuple`

host

Alias for field number 1

module

Alias for field number 0

port

Alias for field number 2

exception `aries_cloudagent.transport.inbound.base.InboundTransportError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.transport.error.TransportError`

Generic inbound transport error.

exception `aries_cloudagent.transport.inbound.base.InboundTransportRegistrationError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.transport.inbound.base.InboundTransportError`

Error in loading an inbound transport.

exception `aries_cloudagent.transport.inbound.base.InboundTransportSetupError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.transport.inbound.base.InboundTransportError`

Setup error for an inbound transport.

aries_cloudagent.transport.inbound.delivery_queue module

The Delivery Queue.

The delivery queue holds and manages messages that have not yet been delivered to their intended destination.

class `aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue`
Bases: `object`

DeliveryQueue class.

Manages undelivered messages.

add_message (`msg: aries_cloudagent.transport.outbound.message.OutboundMessage`)
Add an OutboundMessage to delivery queue.

The message is added once per recipient key

Parameters `msg` – The OutboundMessage to add

expire_messages (`ttl=None`)
Expire messages that are past the time limit.

Parameters `ttl` – Optional. Allows override of configured ttl

get_one_message_for_key (*key: str*)
Remove and return a matching message.

Parameters `key` – The key to use for lookup

has_message_for_key (*key: str*)
Check for queued messages by key.

Parameters `key` – The key to use for lookup

inspect_all_messages_for_key (*key: str*)
Return all messages for key.

Parameters `key` – The key to use for lookup

message_count_for_key (*key: str*)
Count of queued messages by key.

Parameters `key` – The key to use for lookup

remove_message_for_key (*key: str, msg: aries_cloudagent.transport.outbound.message.OutboundMessage*)
Remove specified message from queue for key.

Parameters

- **key** – The key to use for lookup
- **msg** – The message to remove from the queue

class `aries_cloudagent.transport.inbound.delivery_queue.QueuedMessage` (*msg: aries_cloudagent.transport.outbound.message.OutboundMessage*)

Bases: `object`

Wrapper Class for queued messages.

Allows tracking Metadata.

older_than (*compare_timestamp: float*) → bool
Age Comparison.

Allows you to test age as compared to the provided timestamp.

Parameters `compare_timestamp` – The timestamp to compare

aries_cloudagent.transport.inbound.http module

Http Transport classes and functions.

class `aries_cloudagent.transport.inbound.http.HttpTransport` (*host: str, port: int, create_session, **kwargs*)

Bases: `aries_cloudagent.transport.inbound.base.BaseInboundTransport`

Http Transport class.

inbound_message_handler (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e754896a0>*)
Message handler for inbound messages.

Parameters `request` – aiohttp request object

Returns The web response

invite_message_handler (*request:* <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e754896a0>)

Message handler for invites.

Parameters **request** – aiohttp request object

Returns The web response

make_application () → <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e754896a0>

Construct the aiohttp application.

start () → None

Start this transport.

Raises `InboundTransportSetupError` – If there was an error starting the webserver

stop () → None

Stop this transport.

aries_cloudagent.transport.inbound.manager module

Inbound transport manager.

```
class aries_cloudagent.transport.inbound.manager.InboundTransportManager (context:
                                                                    aries_cloudagent.config.in
                                                                    re-
                                                                    ceive_inbound:
                                                                    Corou-
                                                                    tine[T_co,
                                                                    T_contra,
                                                                    V_co],
                                                                    re-
                                                                    turn_inbound:
                                                                    Callable
                                                                    =
                                                                    None)
```

Bases: `object`

Inbound transport manager class.

closed_session (*session:* `aries_cloudagent.transport.inbound.session.InboundSession`)

Clean up a closed session.

Returns an undelivered message to the caller if possible.

create_session (*transport_type:* `str`, ***, *accept_undelivered:* `bool` = `False`,
can_respond: `bool` = `False`, *client_info:* `dict` = `None`, *wire_format:*
`aries_cloudagent.transport.wire_format.BaseWireFormat` = `None`)

Create a new inbound session.

Parameters

- **transport_type** – The inbound transport identifier
- **accept_undelivered** – Flag for accepting undelivered messages
- **can_respond** – Flag indicating that the transport can send responses
- **client_info** – An optional dict describing the client
- **wire_format** – Override the wire format for this session

dispatch_complete (*message: aries_cloudagent.transport.inbound.message.InboundMessage*,
completed: aries_cloudagent.utils.task_queue.CompletedTask)
 Handle completion of message dispatch.

get_transport_instance (*transport_id: str*) → *aries_cloudagent.transport.inbound.base.BaseInboundTransport*
 Get an instance of a running transport by ID.

process_undelivered (*session: aries_cloudagent.transport.inbound.session.InboundSession*)
 Interact with undelivered queue to find applicable messages.

Parameters session – The inbound session

register (*config: aries_cloudagent.transport.inbound.base.InboundTransportConfiguration*) → *str*
 Register transport module.

Parameters config – The inbound transport configuration

register_transport (*transport: aries_cloudagent.transport.inbound.base.BaseInboundTransport*,
transport_id: str) → *str*
 Register a new inbound transport class.

Parameters

- **transport** – Transport instance to register
- **transport_id** – The transport ID to register

return_to_session (*outbound: aries_cloudagent.transport.outbound.message.OutboundMessage*)
 → *bool*
 Return an outbound message via an open session, if possible.

return_undelivered (*outbound: aries_cloudagent.transport.outbound.message.OutboundMessage*)
 → *bool*
 Add an undelivered message to the undelivered queue.

At this point the message could not be associated with an inbound session and could not be delivered via an outbound transport.

setup ()
 Perform setup operations.

start ()
 Start all registered transports.

start_transport (*transport_id: str*)
 Start a registered inbound transport.

Parameters transport_id – ID for the inbound transport to start

stop (*wait: bool = True*)
 Stop all registered transports.

aries_cloudagent.transport.inbound.message module

Classes representing inbound messages.

```
class aries_cloudagent.transport.inbound.message.InboundMessage (payload:
    Union[str,
    bytes], receipt:
    aries_cloudagent.transport.inbound.rec
    *,      connec-
    tion_id:  str
    = None, ses-
    sion_id: str =
    None, trans-
    port_type: str
    = None)
```

Bases: `object`

Container class linking a message payload with its receipt details.

`aries_cloudagent.transport.inbound.receipt` module

Classes for representing message receipt details.

```
class aries_cloudagent.transport.inbound.receipt.MessageReceipt (*,      connec-
    tion_id:  str
    = None, di-
    rect_response_mode:
    str = None,
    in_time: date-
    time.datetime
    =      None,
    raw_message:
    str = None,
    recipi-
    ent_verkey: str
    = None, recipi-
    ent_id: str
    = None, recipi-
    ent_id_public:
    bool = None,
    sender_id:
    str = None,
    sender_verkey:
    str = None,
    thread_id: str
    = None)
```

Bases: `object`

Properties of an agent message's delivery.

REPLY_MODE_ALL = 'all'

REPLY_MODE_NONE = 'none'

REPLY_MODE_THREAD = 'thread'

connection_id

Accessor for the pairwise connection identifier.

Returns This context's connection identifier

direct_response_mode

Accessor for the requested direct response mode.

Returns This context's requested direct response mode

direct_response_requested

Accessor for the the state of the direct response mode.

Returns This context's requested direct response mode

in_time

Accessor for the datetime the message was received.

Returns This context's received time

raw_message

Accessor for the raw message text.

Returns The raw message text

recipient_did

Accessor for the recipient DID which corresponds with the verkey.

Returns The recipient DID

recipient_did_public

Check if the recipient did is public.

Indicates whether the message is associated with a public (ledger) recipient DID.

Returns True if the recipient's DID is public, else false

recipient_verkey

Accessor for the recipient verkey key used to pack the incoming request.

Returns The recipient verkey

sender_did

Accessor for the sender DID which corresponds with the verkey.

Returns The sender did

sender_verkey

Accessor for the sender public key used to pack the incoming request.

Returns This context's sender's verkey

thread_id

Accessor for the identifier of the message thread.

Returns The delivery thread ID

aries_cloudagent.transport.inbound.session module

Inbound connection handling classes.

```
class aries_cloudagent.transport.inbound.session.AcceptResult (accepted: bool,  
retry: bool =  
False)
```

Bases: `object`

Represent the result of `accept_response`.

```
class aries_cloudagent.transport.inbound.session.InboundSession (*,      context:
                                                                    aries_cloudagent.config.injection_cont
                                                                    in-
                                                                    bound_handler:
                                                                    Callable, ses-
                                                                    sion_id:  str,
                                                                    wire_format:
                                                                    aries_cloudagent.transport.wire_forma
                                                                    ac-
                                                                    cept_undelivered:
                                                                    bool = False,
                                                                    can_respond:
                                                                    bool = False,
                                                                    client_info:
                                                                    dict = None,
                                                                    close_handler:
                                                                    Callable =
                                                                    None,      re-
                                                                    ply_mode: str
                                                                    = None, re-
                                                                    ply_thread_ids:
                                                                    Sequence[str]
                                                                    = None, re-
                                                                    ply_verkeys:
                                                                    Sequence[str]
                                                                    = None, trans-
                                                                    port_type: str
                                                                    = None)
```

Bases: `object`

Track an open transport connection for direct routing of outbound messages.

accept_response (*message: aries_cloudagent.transport.outbound.message.OutboundMessage*) →
aries_cloudagent.transport.inbound.session.AcceptResult
Try to queue an outbound message if it applies to this session.

Returns: a tuple of (message buffered, retry later)

add_reply_thread_ids (**thids*)
Add a thread ID to the set of potential reply targets.

add_reply_verkeys (**verkeys*)
Add a verkey to the set of potential reply targets.

can_respond
Accessor for the session can-respond state.

clear_response ()
Handle when the buffered response message has been delivered.

close ()
Setter for the session closed state.

closed
Accessor for the session closed state.

encode_outbound (*outbound: aries_cloudagent.transport.outbound.message.OutboundMessage*) →
aries_cloudagent.transport.outbound.message.OutboundMessage
Apply wire formatting to an outbound message.

parse_inbound (*payload_enc: Union[str, bytes]*) → *aries_cloudagent.transport.inbound.message.InboundMessage*
 Convert a message payload and to an inbound message.

process_inbound (*message: aries_cloudagent.transport.inbound.message.InboundMessage*)
 Process an incoming message and update the session metadata as necessary.

Parameters *message* – The inbound message instance

receive (*payload_enc: Union[str, bytes]*) → *aries_cloudagent.transport.inbound.message.InboundMessage*
 Receive a new message payload and dispatch the message.

receive_inbound (*message: aries_cloudagent.transport.inbound.message.InboundMessage*)
 Deliver the inbound message to the conductor.

reply_mode
 Accessor for the session reply mode.

reply_thread_ids
 Accessor for the reply thread IDs.

reply_verkeys
 Accessor for the reply verkeys.

response_buffered
 Check if a response is currently buffered.

select_outbound (*message: aries_cloudagent.transport.outbound.message.OutboundMessage*) → *bool*
 Determine if an outbound message should be sent to this session.

Parameters *message* – The outbound message to be checked

set_response (*message: aries_cloudagent.transport.outbound.message.OutboundMessage*)
 Set the contents of the response message buffer.

wait_response () → *Union[str, bytes]*
 Wait for a response to be buffered and pack it.

aries_cloudagent.transport.inbound.ws module

Websockets Transport classes and functions.

class *aries_cloudagent.transport.inbound.ws.WsTransport* (*host: str, port: int, create_session, **kwargs*)

Bases: *aries_cloudagent.transport.inbound.base.BaseInboundTransport*

Websockets Transport class.

inbound_message_handler (*request*)
 Message handler for inbound messages.

Parameters *request* – aiohttp request object

Returns The web response

make_application () → *<sphinx.ext.autodoc.importer._MockObject object at 0x7f9e74ce4748>*
 Construct the aiohttp application.

scheme
 Accessor for this transport's scheme.

start () → *None*
 Start this transport.

Raises `InboundTransportSetupError` – If there was an error starting the webserver

stop() → None
Stop this transport.

aries_cloudagent.transport.outbound package

Submodules

aries_cloudagent.transport.outbound.base module

Base outbound transport.

```
class aries_cloudagent.transport.outbound.base.BaseOutboundTransport (wire_format:
                                                                    aries_cloudagent.transport.wire
                                                                    =
                                                                    None)
```

Bases: `abc.ABC`

Base outbound transport class.

collector
Accessor for the stats collector instance.

handle_message (payload: `Union[str, bytes]`, endpoint: `str`)
Handle message from queue.

Parameters

- **payload** – message payload in string or byte format
- **endpoint** – URI endpoint for delivery

start()
Start the transport.

stop()
Shut down the transport.

wire_format
Accessor for a custom wire format for the transport.

```
exception aries_cloudagent.transport.outbound.base.OutboundDeliveryError (*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: `aries_cloudagent.transport.outbound.base.OutboundTransportError`

Base exception when a message cannot be delivered via an outbound transport.

```
exception aries_cloudagent.transport.outbound.base.OutboundTransportError (*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)
```

Bases: `aries_cloudagent.transport.error.TransportError`

Generic outbound transport error.

exception `aries_cloudagent.transport.outbound.base.OutboundTransportRegistrationError` (*args, error_code: str) = None, **kwargs

Bases: `aries_cloudagent.transport.outbound.base.OutboundTransportError`

Outbound transport registration error.

aries_cloudagent.transport.outbound.http module

Http outbound transport.

class `aries_cloudagent.transport.outbound.http.HttpTransport`
Bases: `aries_cloudagent.transport.outbound.base.BaseOutboundTransport`
Http outbound transport class.
handle_message (payload: Union[str, bytes], endpoint: str)
Handle message from queue.
Parameters **message** – *OutboundMessage* to send over transport implementation
schemes = ('http', 'https')
start ()
Start the transport.
stop ()
Stop the transport.

aries_cloudagent.transport.outbound.manager module

Outbound transport manager.

class `aries_cloudagent.transport.outbound.manager.OutboundTransportManager` (context: aries_cloudagent.config.HandlerNotDeliveredCallable) = None
Bases: `object`
Outbound transport manager class.
deliver_queued_message (queued: aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage) → `_asyncio.Task`
Kick off delivery of a queued message.
encode_queued_message (queued: aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage) → `_asyncio.Task`
Kick off encoding of a queued message.

enqueue_message (*context: aries_cloudagent.config.injection_context.InjectionContext, outbound: aries_cloudagent.transport.outbound.message.OutboundMessage*)

Add an outbound message to the queue.

Parameters

- **context** – The context of the request
- **outbound** – The outbound message to deliver

enqueue_webhook (*topic: str, payload: dict, endpoint: str, max_attempts: int = None*)

Add a webhook to the queue.

Parameters

- **topic** – The webhook topic
- **payload** – The webhook payload
- **endpoint** – The webhook endpoint
- **max_attempts** – Override the maximum number of attempts

Raises `OutboundDeliveryError` – if the associated transport is not running

finished_deliver (*queued: aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage, completed: aries_cloudagent.utils.task_queue.CompletedTask*)

Handle completion of queued message delivery.

finished_encode (*queued: aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage, completed: aries_cloudagent.utils.task_queue.CompletedTask*)

Handle completion of queued message encoding.

flush ()

Wait for any queued messages to be delivered.

get_registered_transport_for_scheme (*scheme: str*) → str

Find the registered transport ID for a given scheme.

get_running_transport_for_endpoint (*endpoint: str*)

Find the running transport ID to use for a given endpoint.

get_running_transport_for_scheme (*scheme: str*) → str

Find the running transport ID for a given scheme.

get_transport_instance (*transport_id: str*) → `aries_cloudagent.transport.outbound.base.BaseOutboundTransport`

Get an instance of a running transport by ID.

perform_encode (*queued: aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage*)

Perform message encoding.

process_queued () → `_asyncio.Task`

Start the process to deliver queued messages if necessary.

Returns: the current queue processing task or None

register (*module: str*) → str

Register a new outbound transport by module path.

Parameters **module** – Module name to register

Raises

- `OutboundTransportRegistrationError` – If the imported class cannot be located

- `OutboundTransportRegistrationError` – If the imported class does not specify a `schemes` attribute
- `OutboundTransportRegistrationError` – If the scheme has already been registered

register_class (*transport_class*: `Type[aries_cloudagent.transport.outbound.base.BaseOutboundTransport]`,
transport_id: `str = None`) → `str`

Register a new outbound transport class.

Parameters **transport_class** – Transport class to register

Raises

- `OutboundTransportRegistrationError` – If the imported class does not specify a `schemes` attribute
- `OutboundTransportRegistrationError` – If the scheme has already been registered

setup ()

Perform setup operations.

start ()

Start all transports and feed messages from the queue.

start_transport (*transport_id*: `str`)

Start a registered transport.

stop (*wait*: `bool = True`)

Stop all running transports.

class `aries_cloudagent.transport.outbound.manager.QueuedOutboundMessage` (*context*:
`aries_cloudagent.config.injection_context.InjectionContext`,
message:
`aries_cloudagent.transport.outbound.base.OutboundMessage`,
target:
`aries_cloudagent.connection.Connection`,
transport_id:
`str`)

Bases: `object`

Class representing an outbound message pending delivery.

STATE_DELIVER = `'deliver'`

STATE_DONE = `'done'`

STATE_ENCODE = `'encode'`

STATE_NEW = `'new'`

STATE_PENDING = `'pending'`

STATE_RETRY = `'retry'`

aries_cloudagent.transport.outbound.message module

Outbound message representation.

```

class aries_cloudagent.transport.outbound.message.OutboundMessage(*, connection_id:
    str = None,
    enc_payload:
    Union[str,
    bytes] =
    None, endpoint: str
    = None,
    payload:
    Union[str,
    bytes], reply_session_id:
    str =
    None, reply_thread_id:
    str =
    None, reply_to_verkey:
    str =
    None, reply_from_verkey:
    str = None,
    target:
    aries_cloudagent.connections.model
    = None,
    target_get_list: Sequence[aries_cloudagent.connection
    = None,
    to_session_only:
    bool =
    False)

```

Bases: `object`

Represents an outgoing message.

aries_cloudagent.transport.outbound.ws module

Websockets outbound transport.

```

class aries_cloudagent.transport.outbound.ws.WsTransport
    Bases: aries_cloudagent.transport.outbound.base.BaseOutboundTransport
    Websockets outbound transport class.
    handle_message(payload: Union[str, bytes], endpoint: str)
        Handle message from queue.
        Parameters message – OutboundMessage to send over transport implementation
    schemes = ('ws', 'wss')
    start()
        Start the outbound transport.

```


stop()
Stop the outbound transport.

aries_cloudagent.transport.queue package

Submodules

aries_cloudagent.transport.queue.base module

Abstract message queue.

class aries_cloudagent.transport.queue.base.**BaseMessageQueue**

Bases: `abc.ABC`

Abstract message queue class.

dequeue (*, *timeout: int = None*)

Dequeue a message.

Returns The dequeued message, or None if a timeout occurs

Raises

- `asyncio.CancelledError` if the queue has been stopped
- `asyncio.TimeoutError` if the timeout is reached

enqueue (*message*)

Enqueue a message.

Parameters **message** – The message to add to the end of the queue

Raises `asyncio.CancelledError` if the queue has been stopped

join ()

Wait for the queue to empty.

reset ()

Empty the queue and reset the stop event.

stop ()

Cancel active iteration of the queue.

task_done ()

Indicate that the current task is complete.

aries_cloudagent.transport.queue.basic module

Basic in memory queue.

class aries_cloudagent.transport.queue.basic.**BasicMessageQueue**

Bases: `aries_cloudagent.transport.queue.base.BaseMessageQueue`

Basic in memory queue implementation class.

dequeue (*, *timeout: int = None*)

Dequeue a message.

Returns The dequeued message, or None if a timeout occurs

Raises

- `asyncio.CancelledError` if the queue has been stopped
- `asyncio.TimeoutError` if the timeout is reached

enqueue (*message*)

Enqueue a message.

Parameters **message** – The message to add to the end of the queue

Raises `asyncio.CancelledError` if the queue has been stopped

join ()

Wait for the queue to empty.

make_queue ()

Create the queue instance.

reset ()

Empty the queue and reset the stop event.

stop ()

Cancel active iteration of the queue.

task_done ()

Indicate that the current task is complete.

Submodules

`aries_cloudagent.transport.error` module

Transport-related error classes and codes.

```
exception aries_cloudagent.transport.error.MessageEncodeError (*args,          er-  
                                                                ror_code: str =  
                                                                None, **kwargs)
```

Bases: `aries_cloudagent.transport.error.WireFormatError`

Message encoding error.

```
error_code = 'message_encode_error'
```

```
exception aries_cloudagent.transport.error.MessageParseError (*args, error_code:  
                                                                str = None,  
                                                                **kwargs)
```

Bases: `aries_cloudagent.transport.error.WireFormatError`

Message parse error.

```
error_code = 'message_parse_error'
```

```
exception aries_cloudagent.transport.error.TransportError (*args, error_code: str  
                                                                = None, **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base class for all transport errors.

```
exception aries_cloudagent.transport.error.WireFormatError (*args, error_code: str  
                                                                = None, **kwargs)
```

Bases: `aries_cloudagent.transport.error.TransportError`

Base class for wire-format errors.

aries_cloudagent.transport.pack_format module

Standard packed message format classes.

class aries_cloudagent.transport.pack_format.PackWireFormat

Bases: *aries_cloudagent.transport.wire_format.BaseWireFormat*

Standard DIDComm message parser and serializer.

encode_message (*context: aries_cloudagent.config.injection_context.InjectionContext, message_json: Union[str, bytes], recipient_keys: Sequence[str], routing_keys: Sequence[str], sender_key: str*) → Union[str, bytes]

Encode an outgoing message for transport.

Parameters

- **context** – The injection context for settings and services
- **message_json** – The message body to serialize
- **recipient_keys** – A sequence of recipient verkeys
- **routing_keys** – A sequence of routing verkeys
- **sender_key** – The verification key of the sending agent

Returns The encoded message

Raises MessageEncodeError – If the message could not be encoded

pack (*context: aries_cloudagent.config.injection_context.InjectionContext, message_json: Union[str, bytes], recipient_keys: Sequence[str], routing_keys: Sequence[str], sender_key: str*)

Look up the wallet instance and perform the message pack.

parse_message (*context: aries_cloudagent.config.injection_context.InjectionContext, message_body: Union[str, bytes]*) → Tuple[dict, aries_cloudagent.transport.inbound.receipt.MessageReceipt]

Deserialize an incoming message and further populate the request context.

Parameters

- **context** – The injection context for settings and services
- **message_body** – The body of the message

Returns A tuple of the parsed message and a message receipt instance

Raises

- MessageParseError – If the JSON parsing failed
- MessageParseError – If a wallet is required but can't be located

unpack (*context: aries_cloudagent.config.injection_context.InjectionContext, message_body: Union[str, bytes], receipt: aries_cloudagent.transport.inbound.receipt.MessageReceipt*)

Look up the wallet instance and perform the message unpack.

aries_cloudagent.transport.stats module

aiohttp stats collector support.

class aries_cloudagent.transport.stats.StatsTracer (*collector:*

aries_cloudagent.utils.stats.Collector,
prefix: str)

Bases: sphinx.ext.autodoc.importer._MockObject

Attach hooks to client session events and report statistics.

connection_queued_end (*session, context, params*)
Handle the end of a queued connection.

connection_queued_start (*session, context, params*)
Handle the start of a queued connection.

connection_ready (*session, context, params*)
Handle the end of connection acquisition.

dns_resolvehost_end (*session, context, params*)
Handle the end of a DNS resolution.

dns_resolvehost_start (*session, context, params*)
Handle the start of a DNS resolution.

request_end (*session, context, params*)
Handle the end of request.

request_start (*session, context, params*)
Handle the start of a request.

socket_connect_start (*session, context, params*)
Handle the start of a socket connection.

aries_cloudagent.transport.wire_format module

Abstract wire format classes.

class `aries_cloudagent.transport.wire_format.BaseWireFormat`
Bases: `object`

Abstract messaging wire format.

encode_message (*context: aries_cloudagent.config.injection_context.InjectionContext, message_json: Union[str, bytes], recipient_keys: Sequence[str], routing_keys: Sequence[str], sender_key: str*) → `Union[str, bytes]`
Encode an outgoing message for transport.

Parameters

- **context** – The injection context for settings and services
- **message_json** – The message body to serialize
- **recipient_keys** – A sequence of recipient verkeys
- **routing_keys** – A sequence of routing verkeys
- **sender_key** – The verification key of the sending agent

Returns The encoded message

Raises `MessageEncodeError` – If the message could not be encoded

parse_message (*context: aries_cloudagent.config.injection_context.InjectionContext, message_body: Union[str, bytes]*) → `Tuple[dict, aries_cloudagent.transport.inbound.receive.MessageReceipt]`
Deserialize an incoming message and further populate the request context.

Parameters

- **context** – The injection context for settings and services

- **message_body** – The body of the message

Returns A tuple of the parsed message and a message receipt instance

Raises `MessageParseError` – If the message can't be parsed

class `aries_cloudagent.transport.wire_format.JsonWireFormat`
 Bases: `aries_cloudagent.transport.wire_format.BaseWireFormat`

Unencrypted wire format.

encode_message (*context:* `aries_cloudagent.config.injection_context.InjectionContext`, *message_json:* `Union[str, bytes]`, *recipient_keys:* `Sequence[str]`, *routing_keys:* `Sequence[str]`, *sender_key:* `str`) → `Union[str, bytes]`

Encode an outgoing message for transport.

Parameters

- **context** – The injection context for settings and services
- **message_json** – The message body to serialize
- **recipient_keys** – A sequence of recipient verkeys
- **routing_keys** – A sequence of routing verkeys
- **sender_key** – The verification key of the sending agent

Returns The encoded message

Raises `MessageEncodeError` – If the message could not be encoded

parse_message (*context:* `aries_cloudagent.config.injection_context.InjectionContext`, *message_body:* `Union[str, bytes]`) → `Tuple[dict, aries_cloudagent.transport.inbound.receipt.MessageReceipt]`

Deserialize an incoming message and further populate the request context.

Parameters

- **context** – The injection context for settings and services
- **message_body** – The body of the message

Returns A tuple of the parsed message and a message receipt instance

Raises `MessageParseError` – If the JSON parsing failed

1.1.12 aries_cloudagent.utils package

Submodules

aries_cloudagent.utils.classloader module

The classloader provides utilities to dynamically load classes and modules.

class `aries_cloudagent.utils.classloader.ClassLoader`
 Bases: `object`

Class used to load classes from modules dynamically.

classmethod **load_class** (*class_name:* `str`, *default_module:* `str = None`, *package:* `str = None`)
 Resolve a complete class path (ie. `typing.Dict`) to the class itself.

Parameters

- **class_name** – the class name
- **default_module** – the default module to load, if not part of in the class name
- **package** – the parent package to search for the module

Returns The resolved class

Raises

- *ClassNotFoundError* – If the class could not be resolved at path
- *ModuleLoadError* – If there was an error loading the module

classmethod load_module (*mod_path: str, package: str = None*) → module
Load a module by its absolute path.

Parameters

- **mod_path** – the absolute or relative module path
- **package** – the parent package to search for the module

Returns The resolved module or *None* if the module cannot be found

Raises *ModuleLoadError* – If there was an error loading the module

classmethod load_subclass_of (*base_class: Type[CT_co], mod_path: str, package: str = None*)
Resolve an implementation of a base path within a module.

Parameters

- **base_class** – the base class being implemented
- **mod_path** – the absolute module path
- **package** – the parent package to search for the module

Returns The resolved class

Raises

- *ClassNotFoundError* – If the module or class implementation could not be found
- *ModuleLoadError* – If there was an error loading the module

classmethod scan_subpackages (*package: str*) → Sequence[str]
Return a list of sub-packages defined under a named package.

exception `aries_cloudagent.utils.classloader.ClassNotFoundError` (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.core.error.BaseError*

Class not found error.

exception `aries_cloudagent.utils.classloader.ModuleLoadError` (*args, error_code: str = None, **kwargs)

Bases: *aries_cloudagent.core.error.BaseError*

Module load error.

aries_cloudagent.utils.http module

HTTP utility methods.

exception `aries_cloudagent.utils.http.FetchError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Error raised when an HTTP fetch fails.

`aries_cloudagent.utils.http.fetch` (url: str, *, headers: dict = None, retry: bool = True, max_attempts: int = 5, interval: float = 1.0, backoff: float = 0.25, request_timeout: float = 10.0, connector: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75de5f60> = None, session: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75dfe080> = None, json: bool = False)

Fetch from an HTTP server with automatic retries and timeouts.

Parameters

- **url** – the address to fetch
- **headers** – an optional dict of headers to send
- **retry** – flag to retry the fetch
- **max_attempts** – the maximum number of attempts to make
- **interval** – the interval between retries, in seconds
- **backoff** – the backoff interval, in seconds
- **request_timeout** – the HTTP request timeout, in seconds
- **connector** – an optional existing BaseConnector
- **session** – a shared ClientSession
- **json** – flag to parse the result as JSON

aries_cloudagent.utils.repeat module

Utils for repeating tasks.

class `aries_cloudagent.utils.repeat.RepeatAttempt` (seq: `aries_cloudagent.utils.repeat.RepeatSequence`, index: int = 1)

Bases: `object`

Represents the current iteration in a repeat sequence.

final

Check if this is the last instance in the sequence.

next () → `aries_cloudagent.utils.repeat.RepeatAttempt`
Get the next attempt instance.

next_interval

Calculate the interval before the next attempt.

timeout (interval: float = None)

Create a context manager for timing out an attempt.

```
class aries_cloudagent.utils.repeat.RepeatSequence (limit: int = 0, interval: float = 0.0,  
                                                    backoff: float = 0.0)
```

Bases: `object`

Represents a repetition sequence.

next_interval (*index: int*) → float
Calculate the time before the next attempt.

start () → aries_cloudagent.utils.repeat.RepeatAttempt
Get the first attempt in the sequence.

aries_cloudagent.utils.stats module

Classes for tracking performance and timing.

```
class aries_cloudagent.utils.stats.Collector (*, enabled: bool = True, log_path: str =  
                                           None)
```

Bases: `object`

Collector for a set of statistics.

enabled
Accessor for the collector's enabled property.

extract (*groups: Sequence[str] = None*) → dict
Extract statistics for a specific set of groups.

log (*name: str, duration: float, start: float = None*)
Log an entry in the statistics if the collector is enabled.

mark (**names*)
Make a custom decorator function for adding to the set of groups.

reset ()
Reset the collector's statistics.

results
Accessor for the current set of collected statistics.

timer (**groups*)
Create a new timer attached to this collector.

wrap (*obj, prop_name: Union[str, Sequence[str]], groups: Sequence[str] = None, *, ignore_missing:*
 bool = False)
Wrap a method on a class or class instance.

wrap_coro (*fn, groups: Sequence[str]*)
Wrap a coroutine instance to collect timing statistics on execution.

wrap_fn (*fn, groups: Sequence[str]*)
Wrap a function instance to collect timing statistics on execution.

```
class aries_cloudagent.utils.stats.Stats
```

Bases: `object`

A collection of statistics.

extract (*names: Sequence[str] = None*) → dict
Summarize the stats in a dictionary.

log (*name: str, duration: float*)
Log an entry in the stats.


```
class aries_cloudagent.utils.stats.Timer (collector: aries_cloudagent.utils.stats.Collector,
                                           groups: Sequence[str])
```

Bases: `object`

Timer instance for a running task.

```
classmethod now ()
    Fetch a standard timer value.
```

```
start () → aries_cloudagent.utils.stats.Timer
    Start the timer.
```

```
stop ()
    Stop the timer.
```

aries_cloudagent.utils.task_queue module

Classes for managing a set of asyncio tasks.

```
class aries_cloudagent.utils.task_queue.CompletedTask (task: _asyncio.Task, exc_info:
                                                         Tuple, ident: str = None, timing: dict = None)
```

Bases: `object`

Represent the result of a queued task.

```
class aries_cloudagent.utils.task_queue.PendingTask (coro: Coroutine[T_co, T_contra,
                                                         V_co], complete_hook: Callable = None,
                                                         ident: str = None,
                                                         task_future: _asyncio.Future = None,
                                                         queued_time: float = None)
```

Bases: `object`

Represent a task in the queue.

```
cancel ()
    Cancel the pending task.
```

```
cancelled
    Accessor for the cancelled property.
```

```
task
    Accessor for the task.
```

```
class aries_cloudagent.utils.task_queue.TaskQueue (max_active: int = 0, timed: bool = False,
                                                         trace_fn: Callable = None)
```

Bases: `object`

A class for managing a set of asyncio tasks.

```
add_active (task: _asyncio.Task, task_complete: Callable = None, ident: str = None, timing: dict = None) → _asyncio.Task
    Register an active async task with an optional completion callback.
```

Parameters

- **task** – The asyncio task instance
- **task_complete** – An optional callback to run on completion
- **ident** – A string identifier for the task
- **timing** – An optional dictionary of timing information

add_pending (*pending: aries_cloudagent.utils.task_queue.PendingTask*)

Add a task to the pending queue.

Parameters **pending** – The *PendingTask* to add to the task queue

cancel ()

Cancel any pending or active tasks in the queue.

cancel_pending ()

Cancel any pending tasks in the queue.

cancelled

Accessor for the cancelled property of the queue.

complete (*timeout: float = None, cleanup: bool = True*)

Cancel any pending tasks and wait for, or cancel active tasks.

completed_task (*task: _asyncio.Task, task_complete: Callable, ident: str, timing: dict = None*)

Clean up after a task has completed and run callbacks.

current_active

Accessor for the current number of active tasks in the queue.

current_pending

Accessor for the current number of pending tasks in the queue.

current_size

Accessor for the total number of tasks in the queue.

drain () → *_asyncio.Task*

Start the process to run queued tasks.

flush ()

Wait for any active or pending tasks to be completed.

max_active

Accessor for the maximum number of active tasks in the queue.

put (*coro: Coroutine[T_co, T_contra, V_co], task_complete: Callable = None, ident: str = None*) →

aries_cloudagent.utils.task_queue.PendingTask

Add a new task to the queue, delaying execution if busy.

Parameters

- **coro** – The coroutine to run
- **task_complete** – A callback to run on completion
- **ident** – A string identifier for the task

Returns: a future resolving to the *asyncio* task instance once queued

ready

Accessor for the ready property of the queue.

run (*coro: Coroutine[T_co, T_contra, V_co], task_complete: Callable = None, ident: str = None, timing:*

dict = None) → *_asyncio.Task*

Start executing a coroutine as an *async* task, bypassing the pending queue.

Parameters

- **coro** – The coroutine to run
- **task_complete** – An optional callback to run on completion
- **ident** – A string identifier for the task

- **timing** – An optional dictionary of timing information

Returns: the new asyncio task instance

wait_for (*timeout: float*)

Wait for all queued tasks to complete with a timeout.

`aries_cloudagent.utils.task_queue.coro_ident` (*coro: Coroutine[T_co, T_contra, V_co]*)

Extract an identifier for a coroutine.

`aries_cloudagent.utils.task_queue.coro_timed` (*coro: Coroutine[T_co, T_contra, V_co],
timing: dict*)

Capture timing for a coroutine.

`aries_cloudagent.utils.task_queue.task_exc_info` (*task: _asyncio.Task*)

Extract exception info from an asyncio task.

1.1.13 aries_cloudagent.verifier package

Submodules

aries_cloudagent.verifier.base module

Base Verifier class.

class `aries_cloudagent.verifier.base.BaseVerifier`

Bases: `abc.ABC`

Base class for verifier.

aries_cloudagent.verifier.indy module

Indy verifier implementation.

class `aries_cloudagent.verifier.indy.IndyVerifier` (*wallet*)

Bases: `aries_cloudagent.verifier.base.BaseVerifier`

Indy holder class.

static pre_verify (*pres_req: dict, pres: dict*) -> (`<enum 'PreVerifyResult'>`, `<class 'str'>`)

Check for essential components and tampering in presentation.

Visit encoded attribute values against raw, and predicate bounds, in presentation, cross-reference against presentation request.

Parameters

- **pres_req** – presentation request
- **pres** – corresponding presentation

Returns An instance of `PreVerifyResult` representing the validation result

verify_presentation (*presentation_request, presentation, schemas, credential_definitions*) → `bool`

Verify a presentation.

Parameters

- **presentation_request** – Presentation request data
- **presentation** – Presentation data

- **schemas** – Schema data
- **credential_definitions** – credential definition data

class `aries_cloudagent.verifier.indy.PreVerifyResult`

Bases: `enum.Enum`

Represent the result of `IndyVerifier.pre_verify`.

ENCODING_MISMATCH = 'demonstrates tampering with raw values'

INCOMPLETE = 'missing essential components'

OK = 'ok'

1.1.14 aries_cloudagent.wallet package

Abstract and Indy wallet handling.

Submodules

aries_cloudagent.wallet.base module

Wallet base class.

class `aries_cloudagent.wallet.base.BaseWallet` (*config: dict*)

Bases: `abc.ABC`

Abstract wallet interface.

WALLET_TYPE = `None`

close()

Close previously-opened wallet, removing it if so configured.

create_local_did (*seed: str = None, did: str = None, metadata: dict = None*) → `aries_cloudagent.wallet.base.DIDInfo`

Create and store a new local DID.

Parameters

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

Returns The created *DIDInfo*

create_public_did (*seed: str = None, did: str = None, metadata: dict = {}*) → `aries_cloudagent.wallet.base.DIDInfo`

Create and store a new public DID.

Implicitly flags all other dids as not public.

Parameters

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

Returns The created *DIDInfo*

create_signing_key (*seed: str = None, metadata: dict = None*) → *aries_cloudagent.wallet.base.KeyInfo*
 Create a new public/private signing keypair.

Parameters

- **seed** – Optional seed allowing deterministic key creation
- **metadata** – Optional metadata to store with the keypair

Returns A *KeyInfo* representing the new record

created

Check whether the wallet was created on the last open call.

get_local_did (*did: str*) → *aries_cloudagent.wallet.base.DIDInfo*
 Find info for a local DID.

Parameters **did** – The DID to get info for

Returns A *DIDInfo* instance for the DID

get_local_did_for_verkey (*verkey: str*) → *aries_cloudagent.wallet.base.DIDInfo*
 Resolve a local DID from a verkey.

Parameters **verkey** – Verkey to get DID info for

Returns A *DIDInfo* instance for the DID

get_local_dids () → *Sequence[aries_cloudagent.wallet.base.DIDInfo]*
 Get list of defined local DIDs.

Returns A list of *DIDInfo* instances

get_public_did () → *aries_cloudagent.wallet.base.DIDInfo*
 Retrieve the public did.

Returns The created *DIDInfo*

get_signing_key (*verkey: str*) → *aries_cloudagent.wallet.base.KeyInfo*
 Fetch info for a signing keypair.

Parameters **verkey** – The verification key of the keypair

Returns A *KeyInfo* representing the keypair

handle

Get internal wallet reference.

Returns Defaults to None

name

Accessor for the wallet name.

Returns Defaults to None

open ()

Open wallet, removing and/or creating it if so configured.

opened

Check whether wallet is currently open.

Returns Defaults to False

pack_message (*message: str, to_verkeys: Sequence[str], from_verkey: str = None*) → bytes
 Pack a message for one or more recipients.

Parameters

- **message** – The message to pack
- **to_verkeys** – The verkeys to pack the message for
- **from_verkey** – The sender verkey

Returns The packed message

replace_local_did_metadata (*did: str, metadata: dict*)

Replace the metadata associated with a local DID.

Parameters

- **did** – DID to replace metadata for
- **metadata** – The new metadata

replace_signing_key_metadata (*verkey: str, metadata: dict*)

Replace the metadata associated with a signing keypair.

Parameters

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

set_public_did (*did: str*) → aries_cloudagent.wallet.base.DIDInfo

Assign the public did.

Returns The created *DIDInfo*

sign_message (*message: bytes, from_verkey: str*) → bytes

Sign a message using the private key associated with a given verkey.

Parameters

- **message** – The message to sign
- **from_verkey** – Sign using the private key related to this verkey

Returns The signature

type

Accessor for the wallet type.

Returns Defaults to None

unpack_message (*enc_message: bytes*) -> (<class 'str'>, <class 'str'>, <class 'str'>)

Unpack a message.

Parameters **enc_message** – The encrypted message

Returns (message, from_verkey, to_verkey)

Return type A tuple

verify_message (*message: bytes, signature: bytes, from_verkey: str*) → bool

Verify a signature against the public key of the signer.

Parameters

- **message** – The message to verify
- **signature** – The signature to verify
- **from_verkey** – Verkey to use in verification

Returns True if verified, else False

```
class aries_cloudagent.wallet.base.DIDInfo (did, verkey, metadata)
    Bases: tuple
```

```
    did
        Alias for field number 0
```

```
    metadata
        Alias for field number 2
```

```
    verkey
        Alias for field number 1
```

```
class aries_cloudagent.wallet.base.KeyInfo (verkey, metadata)
    Bases: tuple
```

```
    metadata
        Alias for field number 1
```

```
    verkey
        Alias for field number 0
```

aries_cloudagent.wallet.basic module

In-memory implementation of BaseWallet interface.

```
class aries_cloudagent.wallet.basic.BasicWallet (config: dict = None)
    Bases: aries_cloudagent.wallet.base.BaseWallet
```

In-memory wallet implementation.

```
WALLET_TYPE = 'basic'
```

```
close()
    Not applicable to in-memory wallet.
```

```
create_local_did(seed: str = None, did: str = None, metadata: dict = None) →
    aries_cloudagent.wallet.base.DIDInfo
    Create and store a new local DID.
```

Parameters

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

Returns A *DIDInfo* instance representing the created DID

Raises *WalletDuplicateError* – If the DID already exists in the wallet

```
create_signing_key(seed: str = None, metadata: dict = None) →
    aries_cloudagent.wallet.base.KeyInfo
    Create a new public/private signing keypair.
```

Parameters

- **seed** – Seed to use for signing key
- **metadata** – Optional metadata to store with the keypair

Returns A *KeyInfo* representing the new record

Raises *WalletDuplicateError* – If the resulting verkey already exists in the wallet

created

Check whether the wallet was created on the last open call.

get_local_did (*did: str*) → aries_cloudagent.wallet.base.DIDInfo

Find info for a local DID.

Parameters **did** – The DID to get info for

Returns A *DIDInfo* instance representing the found DID

Raises *WalletNotFoundError* – If the DID is not found

get_local_did_for_verkey (*verkey: str*) → aries_cloudagent.wallet.base.DIDInfo

Resolve a local DID from a verkey.

Parameters **verkey** – The verkey to get the local DID for

Returns A *DIDInfo* instance representing the found DID

Raises *WalletNotFoundError* – If the verkey is not found

get_local_dids () → Sequence[aries_cloudagent.wallet.base.DIDInfo]

Get list of defined local DIDs.

Returns A list of locally stored DIDs as *DIDInfo* instances

get_signing_key (*verkey: str*) → aries_cloudagent.wallet.base.KeyInfo

Fetch info for a signing keypair.

Parameters **verkey** – The verification key of the keypair

Returns A *KeyInfo* representing the keypair

Raises *WalletNotFoundError* – if no keypair is associated with the verification key

name

Accessor for the wallet name.

open ()

Not applicable to in-memory wallet.

opened

Check whether wallet is currently open.

Returns True

pack_message (*message: str, to_verkeys: Sequence[str], from_verkey: str = None*) → bytes

Pack a message for one or more recipients.

Parameters

- **message** – The message to pack
- **to_verkeys** – List of verkeys to pack for
- **from_verkey** – Sender verkey to pack from

Returns The resulting packed message bytes

replace_local_did_metadata (*did: str, metadata: dict*)

Replace metadata for a local DID.

Parameters

- **did** – The DID to replace metadata for
- **metadata** – The new metadata

Raises `WalletNotFoundError` – If the DID doesn't exist

replace_signing_key_metadata (*verkey: str, metadata: dict*)

Replace the metadata associated with a signing keypair.

Parameters

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

Raises `WalletNotFoundError` – if no keypair is associated with the verification key

sign_message (*message: bytes, from_verkey: str*) → bytes

Sign a message using the private key associated with a given verkey.

Parameters

- **message** – Message bytes to sign
- **from_verkey** – The verkey to use to sign

Returns A signature

Raises

- `WalletError` – If the message is not provided
- `WalletError` – If the verkey is not provided

unpack_message (*enc_message: bytes*) -> (<class 'str'>, <class 'str'>, <class 'str'>)

Unpack a message.

Parameters **enc_message** – The packed message bytes

Returns (message, from_verkey, to_verkey)

Return type A tuple

Raises

- `WalletError` – If the message is not provided
- `WalletError` – If there is a problem unpacking the message

verify_message (*message: bytes, signature: bytes, from_verkey: str*) → bool

Verify a signature against the public key of the signer.

Parameters

- **message** – Message to verify
- **signature** – Signature to verify
- **from_verkey** – Verkey to use in verification

Returns True if verified, else False

Raises

- `WalletError` – If the verkey is not provided
- `WalletError` – If the signature is not provided
- `WalletError` – If the message is not provided

aries_cloudagent.wallet.crypto module

Cryptography functions used by BasicWallet.

```
class aries_cloudagent.wallet.crypto.PackMessageSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed message schema.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.wallet.crypto.PackRecipientHeaderSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed recipient header schema.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.wallet.crypto.PackRecipientSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed recipient schema.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.wallet.crypto.PackRecipientsSchema (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
```

Bases: `marshmallow.schema.Schema`

Packed recipients schema.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
aries_cloudagent.wallet.crypto.create_keypair (seed: bytes = None) → Tuple[bytes, bytes]
```

Create a public and private signing keypair from a seed value.

Parameters `seed` – Seed for keypair

Returns A tuple of (public key, secret key)

`aries_cloudagent.wallet.crypto.decode_pack_message` (*enc_message: bytes, find_key: Callable*) → Tuple[str, Optional[str], str]

Decode a packed message.

Disassemble and unencrypt a packed message, returning the message content, verification key of the sender (if available), and verification key of the recipient.

Parameters

- **enc_message** – The encrypted message
- **find_key** – Function to retrieve private key

Returns A tuple of (message, sender_vk, recip_vk)

Raises

- `ValueError` – If the packed message is invalid
- `ValueError` – If the packed message recipients are invalid
- `ValueError` – If the pack algorithm is unsupported
- `ValueError` – If the sender's public key was not provided

`aries_cloudagent.wallet.crypto.decode_pack_message_outer` (*enc_message: bytes*) → Tuple[dict, dict, bool]

Decode the outer wrapper of a packed message and extract the recipients.

Parameters **enc_message** – The encrypted message

Returns: a tuple of the decoded wrapper, recipients, and authcrypt flag

`aries_cloudagent.wallet.crypto.decode_pack_message_payload` (*wrapper: dict, payload_key: bytes*) → str

Decode the payload of a packed message once the CEK is known.

Parameters

- **wrapper** – The decoded message wrapper
- **payload_key** – The decrypted payload key

`aries_cloudagent.wallet.crypto.decrypt_plaintext` (*ciphertext: bytes, recips_bin: bytes, nonce: bytes, key: bytes*) → str

Decrypt the payload of a packed message.

Parameters

- **ciphertext** –
- **recips_bin** –
- **nonce** –
- **key** –

Returns The decrypted string

`aries_cloudagent.wallet.crypto.encode_pack_message` (*message: str, to_verkeys: Sequence[bytes], from_secret: bytes = None*) → bytes

Assemble a packed message for a set of recipients, optionally including the sender.

Parameters

- **message** – The message to pack
- **to_verkeys** – The verkeys to pack the message for
- **from_secret** – The sender secret

Returns The encoded message

`aries_cloudagent.wallet.crypto.encrypt_plaintext` (*message: str, add_data: bytes, key: bytes*) → Tuple[bytes, bytes, bytes]

Encrypt the payload of a packed message.

Parameters

- **message** – Message to encrypt
- **add_data** –
- **key** – Key used for encryption

Returns A tuple of (ciphertext, nonce, tag)

`aries_cloudagent.wallet.crypto.extract_pack_recipients` (*recipients: Sequence[dict]*) → dict

Extract the pack message recipients into a dict indexed by verkey.

Parameters **recipients** – Recipients to locate

Raises `ValueError` – If the recipients block is mal-formatted

`aries_cloudagent.wallet.crypto.extract_payload_key` (*sender_cek: dict, recip_secret: bytes*) → Tuple[bytes, str]

Extract the payload key from pack recipient details.

Returns: A tuple of the CEK and sender verkey

`aries_cloudagent.wallet.crypto.prepare_pack_recipient_keys` (*to_verkeys: Sequence[bytes], from_secret: bytes = None*) → Tuple[str, bytes]

Assemble the recipients block of a packed message.

Parameters

- **to_verkeys** – Verkeys of recipients
- **from_secret** – Secret to use for signing keys

Returns A tuple of (json result, key)

`aries_cloudagent.wallet.crypto.random_seed`() → bytes

Generate a random seed value.

Returns A new random seed

`aries_cloudagent.wallet.crypto.seed_to_did` (*seed: str*) → str

Derive a DID from a seed value.

Parameters **seed** – The seed to derive

Returns The DID derived from the seed

`aries_cloudagent.wallet.crypto.sign_message` (*message: bytes, secret: bytes*) → bytes

Sign a message using a private signing key.

Parameters

- **message** – The message to sign
- **secret** – The private signing key

Returns The signature

`aries_cloudagent.wallet.crypto.sign_pk_from_sk(secret: bytes) → bytes`
 Extract the verkey from a secret signing key.

`aries_cloudagent.wallet.crypto.validate_seed(seed: (<class 'str'>, <class 'bytes'>)) → bytes`
 Convert a seed parameter to standard format and check length.

Parameters **seed** – The seed to validate**Returns** The validated and encoded seed

`aries_cloudagent.wallet.crypto.verify_signed_message(signed: bytes, verkey: bytes) → bool`
 Verify a signed message according to a public verification key.

Parameters

- **signed** – The signed message
- **verkey** – The verkey to use in verification

Returns True if verified, else False**aries_cloudagent.wallet.error module**

Wallet-related exceptions.

exception `aries_cloudagent.wallet.error.WalletDuplicateError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.wallet.error.WalletError`

Duplicate record exception.

exception `aries_cloudagent.wallet.error.WalletError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

General wallet exception.

exception `aries_cloudagent.wallet.error.WalletNotFoundError(*args, error_code: str = None, **kwargs)`

Bases: `aries_cloudagent.wallet.error.WalletError`

Record not found exception.

aries_cloudagent.wallet.indy module

Indy implementation of BaseWallet interface.

class `aries_cloudagent.wallet.indy.IndyWallet(config: dict = None)`

Bases: `aries_cloudagent.wallet.base.BaseWallet`

Indy wallet implementation.

```

DEFAULT_FRESHNESS = 0
DEFAULT_KEY = ''
DEFAULT_KEY_DERIVATION = 'ARGON2I_MOD'
DEFAULT_NAME = 'default'
DEFAULT_STORAGE_TYPE = None
KEY_DERIVATION_ARGON2I_INT = 'ARGON2I_INT'
KEY_DERIVATION_ARGON2I_MOD = 'ARGON2I_MOD'
KEY_DERIVATION_RAW = 'RAW'
WALLET_TYPE = 'indy'

```

close()

Close previously-opened wallet, removing it if so configured.

create (*replace: bool = False*)

Create a new wallet.

Parameters **replace** – Removes the old wallet if True

Raises

- `WalletError` – If there was a problem removing the wallet
- `WalletError` – IF there was a libindy error

create_local_did (*seed: str = None, did: str = None, metadata: dict = None*) → `aries_cloudagent.wallet.base.DIDInfo`

Create and store a new local DID.

Parameters

- **seed** – Optional seed to use for did
- **did** – The DID to use
- **metadata** – Metadata to store with DID

Returns A *DIDInfo* instance representing the created DID

Raises

- `WalletDuplicateError` – If the DID already exists in the wallet
- `WalletError` – If there is a libindy error

create_signing_key (*seed: str = None, metadata: dict = None*) → `aries_cloudagent.wallet.base.KeyInfo`

Create a new public/private signing keypair.

Parameters

- **seed** – Seed for key
- **metadata** – Optional metadata to store with the keypair

Returns A *KeyInfo* representing the new record

Raises

- `WalletDuplicateError` – If the resulting verkey already exists in the wallet
- `WalletError` – If there is a libindy error

created

Check whether the wallet was created on the last open call.

classmethod generate_wallet_key (*seed: str = None*) → str

Generate a raw Indy wallet key.

get_credential_definition_tag_policy (*credential_definition_id: str*)

Return the tag policy for a given credential definition ID.

get_local_did (*did: str*) → aries_cloudagent.wallet.base.DIDInfo

Find info for a local DID.

Parameters *did* – The DID to get info for

Returns A *DIDInfo* instance representing the found DID

Raises

- *WalletNotFoundError* – If the DID is not found
- *WalletError* – If there is a libindy error

get_local_did_for_verkey (*verkey: str*) → aries_cloudagent.wallet.base.DIDInfo

Resolve a local DID from a verkey.

Parameters *verkey* – The verkey to get the local DID for

Returns A *DIDInfo* instance representing the found DID

Raises *WalletNotFoundError* – If the verkey is not found

get_local_dids () → Sequence[aries_cloudagent.wallet.base.DIDInfo]

Get list of defined local DIDs.

Returns A list of locally stored DIDs as *DIDInfo* instances

get_signing_key (*verkey: str*) → aries_cloudagent.wallet.base.KeyInfo

Fetch info for a signing keypair.

Parameters *verkey* – The verification key of the keypair

Returns A *KeyInfo* representing the keypair

Raises

- *WalletNotFoundError* – If no keypair is associated with the verification key
- *WalletError* – If there is a libindy error

handle

Get internal wallet reference.

Returns A handle to the wallet

master_secret_id

Accessor for the master secret id.

Returns The master secret id

name

Accessor for the wallet name.

Returns The wallet name

open ()

Open wallet, removing and/or creating it if so configured.

Raises

- `WalletError` – If wallet not found after creation
- `WalletNotFoundError` – If the wallet is not found
- `WalletError` – If the wallet is already open
- `WalletError` – If there is a libindy error

opened

Check whether wallet is currently open.

Returns True if open, else False

pack_message (*message: str, to_verkeys: Sequence[str], from_verkey: str = None*) → bytes

Pack a message for one or more recipients.

Parameters

- **message** – The message to pack
- **to_verkeys** – List of verkeys to pack for
- **from_verkey** – Sender verkey to pack from

Returns The resulting packed message bytes

Raises

- `WalletError` – If no message is provided
- `WalletError` – If a libindy error occurs

remove ()

Remove an existing wallet.

Raises

- `WalletNotFoundError` – If the wallet could not be found
- `WalletError` – If there was an libindy error

replace_local_did_metadata (*did: str, metadata: dict*)

Replace metadata for a local DID.

Parameters

- **did** – The DID to replace metadata for
- **metadata** – The new metadata

replace_signing_key_metadata (*verkey: str, metadata: dict*)

Replace the metadata associated with a signing keypair.

Parameters

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

Raises `WalletNotFoundError` – if no keypair is associated with the verification key

set_credential_definition_tag_policy (*credential_definition_id: str, taggables: Sequence[str] = None, retroactive: bool = True*)

Set the tag policy for a given credential definition ID.

Parameters

- **credential_definition_id** – The ID of the credential definition
- **taggables** – A sequence of string values representing attribute names

- **retroactive** – Whether to apply the policy to previously-stored credentials

sign_message (*message: bytes, from_verkey: str*) → bytes

Sign a message using the private key associated with a given verkey.

Parameters

- **message** – Message bytes to sign
- **from_verkey** – The verkey to use to sign

Returns A signature

Raises

- `WalletError` – If the message is not provided
- `WalletError` – If the verkey is not provided
- `WalletError` – If a libindy error occurs

unpack_message (*enc_message: bytes*) -> (<class 'str'>, <class 'str'>, <class 'str'>)

Unpack a message.

Parameters **enc_message** – The packed message bytes

Returns (message, from_verkey, to_verkey)

Return type A tuple

Raises

- `WalletError` – If the message is not provided
- `WalletError` – If a libindy error occurs

verify_message (*message: bytes, signature: bytes, from_verkey: str*) → bool

Verify a signature against the public key of the signer.

Parameters

- **message** – Message to verify
- **signature** – Signature to verify
- **from_verkey** – Verkey to use in verification

Returns True if verified, else False

Raises

- `WalletError` – If the verkey is not provided
- `WalletError` – If the signature is not provided
- `WalletError` – If the message is not provided
- `WalletError` – If a libindy error occurs

aries_cloudagent.wallet.plugin module

Utility for loading Postgres wallet plug-in.

`aries_cloudagent.wallet.plugin.file_ext()`

Determine file extension based on platform.

```
aries_cloudagent.wallet.plugin.load_postgres_plugin(storage_config, storage_creds,
                                                    raise_exc=False)
```

Load postgres dll and configure postgres wallet.

aries_cloudagent.wallet.provider module

Default wallet provider classes.

```
class aries_cloudagent.wallet.provider.WalletProvider
```

Bases: *aries_cloudagent.config.base.BaseProvider*

Provider for the default configurable wallet classes.

```
WALLET_TYPES = {'basic': 'aries_cloudagent.wallet.basic.BasicWallet', 'indy': 'aries
```

```
provide (settings: aries_cloudagent.config.base.BaseSettings, injector:
          aries_cloudagent.config.base.BaseInjector)
    Create and open the wallet instance.
```

aries_cloudagent.wallet.routes module

Wallet admin routes.

```
class aries_cloudagent.wallet.routes.DIDListSchema(*, only=None, exclude=(),
                                                    many=False, context=None,
                                                    load_only=(), dump_only=(),
                                                    partial=False, unknown=None)
```

Bases: *marshmallow.schema.Schema*

Result schema for connection list.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.wallet.routes.DIDResultSchema(*, only=None, exclude=(), many=False, con-
                                                    text=None, load_only=(),
                                                    dump_only=(), partial=False,
                                                    unknown=None)
```

Bases: *marshmallow.schema.Schema*

Result schema for a DID.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.wallet.routes.DIDSchema(*, only=None, exclude=(), many=False,
                                                    context=None, load_only=(),
                                                    dump_only=(), partial=False, un-
                                                    known=None)
```

Bases: *marshmallow.schema.Schema*

Result schema for a DID.

```
opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.wallet.routes.GetTagPolicyResultSchema (*,    only=None,
                                                                exclude=(),
                                                                many=False,
                                                                context=None,
                                                                load_only=(),
                                                                dump_only=(),
                                                                partial=False,
                                                                unknown=None)
```

Bases: `marshmallow.schema.Schema`

Result schema for tagging policy get request.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.wallet.routes.SetTagPolicyRequestSchema (*,    only=None,
                                                                exclude=(),
                                                                many=False,
                                                                context=None,
                                                                load_only=(),
                                                                dump_only=(),
                                                                par-
                                                                tial=False,    un-
                                                                known=None)
```

Bases: `marshmallow.schema.Schema`

Request schema for tagging policy set request.

opts = `<marshmallow.schema.SchemaOpts object>`

`aries_cloudagent.wallet.routes.format_did_info` (*info: aries_cloudagent.wallet.base.DIDInfo*)
Serialize a DIDInfo object.

`aries_cloudagent.wallet.routes.register` (*app: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75024a58>*)

Register routes.

`aries_cloudagent.wallet.routes.wallet_create_did` (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75024a58>*)

Request handler for creating a new wallet DID.

Parameters **request** – aiohttp request object

Returns The DID info

`aries_cloudagent.wallet.routes.wallet_did_list` (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75024a58>*)

Request handler for searching wallet DIDs.

Parameters **request** – aiohttp request object

Returns The DID list response

`aries_cloudagent.wallet.routes.wallet_get_public_did` (*request: <sphinx.ext.autodoc.importer._MockObject object at 0x7f9e75024a58>*)

Request handler for fetching the current public DID.

Parameters **request** – aiohttp request object

Returns The DID info

```
aries_cloudagent.wallet.routes.wallet_get_tagging_policy(request:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at
                                                         0x7f9e75024a58>)
```

Request handler for getting the tag policy associated with a cred def.

Parameters **request** – aiohttp request object

Returns A JSON object containing the tagging policy

```
aries_cloudagent.wallet.routes.wallet_set_public_did(request:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at 0x7f9e75024a58>)
```

Request handler for setting the current public DID.

Parameters **request** – aiohttp request object

Returns The updated DID info

```
aries_cloudagent.wallet.routes.wallet_set_tagging_policy(request:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at
                                                         0x7f9e75024a58>)
```

Request handler for setting the tag policy associated with a cred def.

Parameters **request** – aiohttp request object

Returns An empty JSON response

aries_cloudagent.wallet.util module

Wallet utility functions.

```
aries_cloudagent.wallet.util.b58_to_bytes(val: str) → bytes
    Convert a base 58 string to bytes.
```

```
aries_cloudagent.wallet.util.b64_to_bytes(val: str, urlsafe=False) → bytes
    Convert a base 64 string to bytes.
```

```
aries_cloudagent.wallet.util.b64_to_str(val: str, urlsafe=False, encoding=None) → str
    Convert a base 64 string to string on input encoding (default utf-8).
```

```
aries_cloudagent.wallet.util.bytes_to_b58(val: bytes) → str
    Convert a byte string to base 58.
```

```
aries_cloudagent.wallet.util.bytes_to_b64(val: bytes, urlsafe=False, pad=True) → str
    Convert a byte string to base 64.
```

```
aries_cloudagent.wallet.util.pad(val: str) → str
    Pad base64 values if need be: JWT calls to omit trailing padding.
```

```
aries_cloudagent.wallet.util.set_urlsafe_b64(val: str, urlsafe: bool = True) → str
    Set URL safety in base64 encoding.
```

```
aries_cloudagent.wallet.util.str_to_b64(val: str, urlsafe=False, encoding=None,
                                         pad=True) → str
    Convert a string to base64 string on input encoding (default utf-8).
```

```
aries_cloudagent.wallet.util.unpad(val: str) → str
    Remove padding from base64 values if need be.
```

1.2 Submodules

1.3 aries_cloudagent.version module

Library version information.

aries_cloudagent.connections package

2.1 Subpackages

2.1.1 aries_cloudagent.connections.models package

Subpackages

aries_cloudagent.connections.models.diddoc package

DID Document model support.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

class aries_cloudagent.connections.models.diddoc.DIDDoc (did: str = None)

Bases: object

DID document, grouping a DID with verification keys and services.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

CONTEXT = 'https://w3id.org/did/v1'

add_service_pubkeys (service: dict, tags: Union[Sequence[str], str]) → List[aries_cloudagent.connections.models.diddoc.publickey.PublicKey]

Add public keys specified in service. Return public keys so discovered.

Parameters

- **service** – service from DID document
- **tags** – potential tags marking public keys of type of interest (the standard is still coalescing)

Raises `ValueError` – for public key reference not present in DID document.

Returns: list of public keys from the document service specification

authnkey

Accessor for public keys marked as authentication keys, by identifier.

classmethod **deserialize** (*did_doc: dict*) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`
Construct DIDDoc object from dict representation.

Parameters **did_doc** – DIDDoc dict representation

Raises `ValueError` – for bad DID or missing mandatory item.

Returns: DIDDoc from input json

did

Accessor for DID.

classmethod **from_json** (*did_doc_json: str*) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`
Construct DIDDoc object from json representation.

Parameters **did_doc_json** – DIDDoc json representation

Returns: DIDDoc from input json

pubkey

Accessor for public keys by identifier.

serialize () → `str`
Dump current object to a JSON-compatible dictionary.

Returns dict representation of current DIDDoc

service

Accessor for services by identifier.

set (*item: Union[aries_cloudagent.connections.models.diddoc.service.Service, aries_cloudagent.connections.models.diddoc.publickey.PublicKey]*) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`
Add or replace service or public key; return current DIDDoc.

Raises `ValueError` – if input item is neither service nor public key.

Parameters **item** – service or public key to set

Returns: the current DIDDoc

to_json () → `str`
Dump current object as json (JSON-LD).

Returns json representation of current DIDDoc

class `aries_cloudagent.connections.models.diddoc.LinkedDataKeySpec` (*ver_type, au-
thn_type, specifier*)

Bases: `tuple`

authn_type

Alias for field number 1

specifier

Alias for field number 2

ver_type

Alias for field number 0

```

class aries_cloudagent.connections.models.diddoc.PublicKey (did: str, ident: str,
                                                            value: str, pk_type:
                                                            aries_cloudagent.connections.models.diddoc.p
                                                            = None, controller:
                                                            str = None, authn:
                                                            bool = False)

```

Bases: `object`

Public key specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

authn

Accessor for the authentication marker.

Returns: whether public key is marked as having DID authentication privilege

controller

Accessor for the controller DID.

did

Accessor for the DID.

id

Accessor for the public key identifier.

to_dict () → dict

Return dict representation of public key to embed in DID document.

type

Accessor for the public key type.

value

Accessor for the public key value.

```

class aries_cloudagent.connections.models.diddoc.PublicKeyType

```

Bases: `enum.Enum`

Class encapsulating public key types.

ED25519_SIG_2018 = `LinkedDataKeySpec(ver_type='Ed25519VerificationKey2018', authn_type=`**EDDSA_SA_SIG_SECP256K1** = `LinkedDataKeySpec(ver_type='Secp256k1VerificationKey2018', au`**RSA_SIG_2018** = `LinkedDataKeySpec(ver_type='RsaVerificationKey2018', authn_type='RsaSig`**authn_type**

Accessor for the authentication type identifier.

get = `<function PublicKeyType.get>`**specification** (val: str) → str

Return specifier and input value for use in public key specification.

Parameters **val** – value of public key

Returns: dict mapping applicable specifier to input value

specifier

Accessor for the value specifier.

ver_type

Accessor for the verification type identifier.

```
class aries_cloudagent.connections.models.diddoc.Service(did: str, ident: str,  
                                                    typ: str, recip_keys:  
                                                    Union[Sequence[T_co],  
                                                    aries_cloudagent.connections.models.diddoc.publ  
                                                    routing_keys:  
                                                    Union[Sequence[T_co],  
                                                    aries_cloudagent.connections.models.diddoc.publ  
                                                    endpoint: str, priority:  
                                                    int = 0)
```

Bases: `object`

Service specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

did

Accessor for the DID value.

endpoint

Accessor for the endpoint value.

id

Accessor for the service identifier.

priority

Accessor for the priority value.

recip_keys

Accessor for the recipient keys.

routing_keys

Accessor for the routing keys.

to_dict () → dict

Return dict representation of service to embed in DID document.

type

Accessor for the service type.

Submodules

aries_cloudagent.connections.models.diddoc.diddoc module

DID Document classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc(did: str = None)
```

Bases: `object`

DID document, grouping a DID with verification keys and services.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

CONTEXT = 'https://w3id.org/did/v1'

```
add_service_pubkeys(service: dict, tags: Union[Sequence[str], str]) → List[aries_cloudagent.connections.models.diddoc.publickey.PublicKey]
```

Add public keys specified in service. Return public keys so discovered.

Parameters

- **service** – service from DID document
- **tags** – potential tags marking public keys of type of interest (the standard is still coalescing)

Raises `ValueError` – for public key reference not present in DID document.

Returns: list of public keys from the document service specification

authnkey

Accessor for public keys marked as authentication keys, by identifier.

```
classmethod deserialize(did_doc: dict) → aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
```

Construct DIDDoc object from dict representation.

Parameters *did_doc* – DIDDoc dict representation

Raises `ValueError` – for bad DID or missing mandatory item.

Returns: DIDDoc from input json

did

Accessor for DID.

```
classmethod from_json(did_doc_json: str) → aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
```

Construct DIDDoc object from json representation.

Parameters *did_doc_json* – DIDDoc json representation

Returns: DIDDoc from input json

pubkey

Accessor for public keys by identifier.

```
serialize() → str
```

Dump current object to a JSON-compatible dictionary.

Returns dict representation of current DIDDoc

service

Accessor for services by identifier.

```
set (item: Union[aries_cloudagent.connections.models.diddoc.service.Service,
    aries_cloudagent.connections.models.diddoc.publickey.PublicKey]) →
    aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
    Add or replace service or public key; return current DIDDoc.
```

Raises `ValueError` – if input item is neither service nor public key.

Parameters `item` – service or public key to set

Returns: the current DIDDoc

```
to_json () → str
    Dump current object as json (JSON-LD).
```

Returns json representation of current DIDDoc

aries_cloudagent.connections.models.diddoc.publickey module

DID Document Public Key classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.publickey.LinkedDataKeySpec (ver_type,
    au-
    thn_type,
    spec-
    i-
    fier)
```

Bases: `tuple`

authn_type

Alias for field number 1

specifier

Alias for field number 2

ver_type

Alias for field number 0

```

class aries_cloudagent.connections.models.diddoc.publickey.PublicKey (did:
                                                                    str,
                                                                    ident:
                                                                    str,
                                                                    value:
                                                                    str,
                                                                    pk_type:
                                                                    aries_cloudagent.connections.m
                                                                    =
                                                                    None,
                                                                    con-
                                                                    troller:
                                                                    str =
                                                                    None,
                                                                    authn:
                                                                    bool =
                                                                    False)

```

Bases: `object`

Public key specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

authn

Accessor for the authentication marker.

Returns: whether public key is marked as having DID authentication privilege

controller

Accessor for the controller DID.

did

Accessor for the DID.

id

Accessor for the public key identifier.

to_dict () → dict

Return dict representation of public key to embed in DID document.

type

Accessor for the public key type.

value

Accessor for the public key value.

```

class aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType

```

Bases: `enum.Enum`

Class encapsulating public key types.

```
ED25519_SIG_2018 = LinkedDataKeySpec(ver_type='Ed25519VerificationKey2018', authn_type=
```

```
EDDSA_SA_SIG_SECP256K1 = LinkedDataKeySpec(ver_type='Secp256k1VerificationKey2018', au
```

```
RSA_SIG_2018 = LinkedDataKeySpec(ver_type='RsaVerificationKey2018', authn_type='RsaSig
```

authn_type

Accessor for the authentication type identifier.

```
get = <function PublicKeyType.get>
```

specification (*val: str*) → str

Return specifier and input value for use in public key specification.

Parameters **val** – value of public key

Returns: dict mapping applicable specifier to input value

specifier

Accessor for the value specifier.

ver_type

Accessor for the verification type identifier.

aries_cloudagent.connections.models.diddoc.service module

DID Document Service classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.service.Service (did: str,  
ident: str,  
typ: str,  
recip_keys:  
Union[Sequence[T_co],  
aries_cloudagent.connections.models.  
routing_keys:  
Union[Sequence[T_co],  
aries_cloudagent.connections.models.  
endpoint: str,  
priority: int  
= 0)
```

Bases: `object`

Service specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

did

Accessor for the DID value.

endpoint

Accessor for the endpoint value.

id

Accessor for the service identifier.

priority

Accessor for the priority value.

recip_keys

Accessor for the recipient keys.

routing_keys

Accessor for the routing keys.

to_dict() → dict

Return dict representation of service to embed in DID document.

type

Accessor for the service type.

aries_cloudagent.connections.models.diddoc.util module

DIDDoc utility methods.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

`aries_cloudagent.connections.models.diddoc.util.canon_did(uri: str) → str`

Convert a URI into a DID if need be, left-stripping ‘did:sov:’ if present.

Parameters `uri` – input URI or DID

Raises `ValueError` – for invalid input.

`aries_cloudagent.connections.models.diddoc.util.canon_ref(did: str, ref: str, delimiter: str = None)`

Given a reference in a DID document, return it in its canonical form of a URI.

Parameters

- **did** – DID acting as the identifier of the DID document
- **ref** – reference to canonicalize, either a DID or a fragment pointing to a location in the DID doc
- **delimiter** – delimiter character marking fragment (default ‘#’) or introducing identifier (‘;’) against DID resource

`aries_cloudagent.connections.models.diddoc.util.ok_did(token: str) → bool`

Whether input token looks like a valid decentralized identifier.

Parameters `token` – candidate string

Returns: whether input token looks like a valid schema identifier

`aries_cloudagent.connections.models.diddoc.util.resource(ref: str, delimiter: str = None) → str`

Extract the resource for an identifier.

Given a (URI) reference, return up to its delimiter (exclusively), or all of it if there is none.

Parameters

- **ref** – reference
- **delimiter** – delimiter character (default None maps to ‘#’, or ‘;’ introduces identifiers)

Submodules

`aries_cloudagent.connections.models.connection_record` module

Handle connection information interface with non-secrets storage.


```
class aries_cloudagent.connections.models.connection_record.ConnectionRecord(*,
    connection_id: str
    = None,
    my_did: str
    = None,
    their_did: str
    = None,
    their_label: str
    = None,
    their_role: str
    = None,
    initiator: str
    = None,
    invitation_key: str
    = None,
    request_id: str
    = None,
    state: str
    = None,
    inbound_connection_id: str
    = None,
    error_msg: str
    = None,
    routing_state: str
    = None,
```

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecord*

Represents a single pairwise connection.

```
ACCEPT_AUTO = 'auto'
ACCEPT_MANUAL = 'manual'
CACHE_ENABLED = True
DIRECTION_RECEIVED = 'received'
DIRECTION_SENT = 'sent'
INITIATOR_EXTERNAL = 'external'
INITIATOR_MULTIUSE = 'multiuse'
INITIATOR_SELF = 'self'
INVITATION_MODE_MULTI = 'multi'
INVITATION_MODE_ONCE = 'once'
INVITATION_MODE_STATIC = 'static'
LOG_STATE_FLAG = 'debug.connections'

class Meta
    Bases: object
    ConnectionRecord metadata.
    schema_class = 'ConnectionRecordSchema'
RECORD_ID_NAME = 'connection_id'
RECORD_TYPE = 'connection'
RECORD_TYPE_INVITATION = 'connection_invitation'
RECORD_TYPE_REQUEST = 'connection_request'
ROUTING_STATE_ACTIVE = 'active'
ROUTING_STATE_ERROR = 'error'
ROUTING_STATE_NONE = 'none'
ROUTING_STATE_REQUEST = 'request'
STATE_ACTIVE = 'active'
STATE_ERROR = 'error'
STATE_INACTIVE = 'inactive'
STATE_INIT = 'init'
STATE_INVITATION = 'invitation'
STATE_REQUEST = 'request'
STATE_RESPONSE = 'response'
TAG_NAMES = {'invitation_key', 'my_did', 'request_id', 'their_did'}
WEBHOOK_TOPIC = 'connections'
```

attach_invitation (*context: aries_cloudagent.config.injection_context.InjectionContext, invitation: aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation*)
Persist the related connection invitation to storage.

Parameters

- **context** – The injection context to use
- **invitation** – The invitation to relate to this connection record

attach_request (*context: aries_cloudagent.config.injection_context.InjectionContext, request: aries_cloudagent.protocols.connections.messages.connection_request.ConnectionRequest*)
Persist the related connection request to storage.

Parameters

- **context** – The injection context to use
- **request** – The request to relate to this connection record

connection_id
Accessor for the ID associated with this connection.

is_multiuse_invitation
Accessor for multi use invitation mode.

is_ready
Accessor for connection readiness.

post_save (*context: aries_cloudagent.config.injection_context.InjectionContext, *args, **kwargs*)
Perform post-save actions.

Parameters context – The injection context to use

record_value
Accessor to for the JSON record value properties for this connection.

classmethod retrieve_by_did (*context: aries_cloudagent.config.injection_context.InjectionContext, their_did: str = None, my_did: str = None, initiator: str = None*) → *aries_cloudagent.connections.models.connection_record.ConnectionRecord*
Retrieve a connection record by target DID.

Parameters

- **context** – The injection context to use
- **their_did** – The target DID to filter by
- **my_did** – One of our DIDs to filter by
- **initiator** – Filter connections by the initiator value

classmethod retrieve_by_invitation_key (*context: aries_cloudagent.config.injection_context.InjectionContext, invitation_key: str, initiator: str = None*) → *aries_cloudagent.connections.models.connection_record.ConnectionRecord*
Retrieve a connection record by invitation key.

Parameters

- **context** – The injection context to use
- **invitation_key** – The key on the originating invitation
- **initiator** – Filter by the initiator value

```
classmethod retrieve_by_request_id (context: aries_cloudagent.config.injection_context.InjectionContext,  
                                request_id: str) →  
                                aries_cloudagent.connections.models.connection_record.ConnectionRecord
```

Retrieve a connection record from our previous request ID.

Parameters

- **context** – The injection context to use
- **request_id** – The ID of the originating connection request

```
retrieve_invitation (context: aries_cloudagent.config.injection_context.InjectionContext) →  
                    aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation
```

Retrieve the related connection invitation.

Parameters context – The injection context to use

```
retrieve_request (context: aries_cloudagent.config.injection_context.InjectionContext) →  
                  aries_cloudagent.protocols.connections.messages.connection_request.ConnectionRequest
```

Retrieve the related connection invitation.

Parameters context – The injection context to use

```
class aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema (*args,  
                                          **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecordSchema*

Schema to allow serialization/deserialization of connection records.

```
class Meta
```

Bases: *object*

ConnectionRecordSchema metadata.

```
model_class
```

alias of *ConnectionRecord*

```
accept = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf  
alias = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r  
connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=  
error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=Non  
inbound_connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, v  
initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<On  
invitation_key = <fields.String(default=<marshmallow.missing>, attribute=None, validat  
invitation_mode = <fields.String(default=<marshmallow.missing>, attribute=None, valid  
my_did = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries  
request_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=No  
routing_state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=  
their_did = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar  
their_label = <fields.String(default=<marshmallow.missing>, attribute=None, validate=N  
their_role = <fields.String(default=<marshmallow.missing>, attribute=None, validate=No
```

aries_cloudagent.connections.models.connection_target module

Record used to handle routing of messages to another agent.

```
class aries_cloudagent.connections.models.connection_target.ConnectionTarget (*,
    did:
    str
    =
    None,
    end-
    point:
    str
    =
    None,
    la-
    bel:
    str
    =
    None,
    re-
    cip-
    i-
    ent_keys:
    Se-
    quence[str]
    =
    None,
    rout-
    ing_keys:
    Se-
    quence[str]
    =
    None,
    sender_key:
    str
    =
    None)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Record used to handle routing of messages to another agent.

```
class Meta
    Bases: object
    ConnectionTarget metadata.
    schema_class = 'ConnectionTargetSchema'
```

```
class aries_cloudagent.connections.models.connection_target.ConnectionTargetSchema (*args,
    **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

ConnectionTarget schema.

```
class Meta
    Bases: object
    ConnectionTargetSchema metadata.
```

model_class
alias of *ConnectionTarget*

did
Used by autodoc_mock_imports.

endpoint
Used by autodoc_mock_imports.

label
Used by autodoc_mock_imports.

recipient_keys
Used by autodoc_mock_imports.

routing_keys
Used by autodoc_mock_imports.

sender_key
Used by autodoc_mock_imports.

aries_cloudagent.protocols package

3.1 Subpackages

3.1.1 aries_cloudagent.protocols.actionmenu package

Subpackages

aries_cloudagent.protocols.actionmenu.handlers package

Submodules

aries_cloudagent.protocols.actionmenu.handlers.menu_handler module

Action menu message handler.

```
class aries_cloudagent.protocols.actionmenu.handlers.menu_handler.MenuHandler  
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for action menus.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:  
        aries_cloudagent.messaging.responder.BaseResponder)  
    Message handler logic for action menus.
```

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.actionmenu.handlers.menu_request_handler module

Action menu request message handler.

```
class aries_cloudagent.protocols.actionmenu.handlers.menu_request_handler.MenuRequestHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for action menu requests.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for action menu requests.
```

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.actionmenu.handlers.perform_handler module

Action menu perform request message handler.

```
class aries_cloudagent.protocols.actionmenu.handlers.perform_handler.PerformHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for action menu perform requests.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for action menu perform requests.
```

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.actionmenu.messages package

Submodules

aries_cloudagent.protocols.actionmenu.messages.menu module

Represents an action menu.

```
class aries_cloudagent.protocols.actionmenu.messages.menu.Menu(*, title: str =
                                                                None, descrip-
                                                                tion: str =
                                                                None, errormsg:
                                                                str = None,
                                                                options: Sequence[aries_cloudagent.protocols.actionmenu.messages.menu.MenuOption] =
                                                                None,
                                                                **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing an action menu.

```
class Meta
```

Bases: *object*

Metadata for an action menu.


```

    handler_class = 'aries_cloudagent.protocols.actionmenu.handlers.menu_handler.MenuHandler'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/action-menu/1.0/menu'
    schema_class = 'MenuSchema'

class aries_cloudagent.protocols.actionmenu.messages.menu.MenuSchema(*args,
                                                                    **kwargs)
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Menu schema class.

    class Meta
        Bases: object
        Menu schema metadata.

        model_class
            alias of Menu

    description = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    errormsg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    options = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, required=False)>
    title = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, required=False)>

```

aries_cloudagent.protocols.actionmenu.messages.menu_request module

Represents a request for an action menu.

```

class aries_cloudagent.protocols.actionmenu.messages.menu_request.MenuRequest(**kwargs)
    Bases: aries_cloudagent.messaging.agent_message.AgentMessage
    Class representing a request for an action menu.

    class Meta
        Bases: object
        Metadata for action menu request.

        handler_class = 'aries_cloudagent.protocols.actionmenu.handlers.menu_request_handler.MenuRequestHandler'
        message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/action-menu/1.0/menu-request'
        schema_class = 'MenuRequestSchema'

class aries_cloudagent.protocols.actionmenu.messages.menu_request.MenuRequestSchema(*args,
                                                                                      **kwargs)
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    MenuRequest schema class.

    class Meta
        Bases: object
        MenuRequest schema metadata.

        model_class
            alias of MenuRequest

```

aries_cloudagent.protocols.actionmenu.messages.perform module

Represents a request to perform a menu action.

```
class aries_cloudagent.protocols.actionmenu.messages.perform.Perform(*,  
                                                                    name:  
                                                                    str =  
                                                                    None,  
                                                                    params:  
                                                                    Map-  
                                                                    ping[str,  
                                                                    str] =  
                                                                    None,  
                                                                    **kwargs)  
  
Bases: aries_cloudagent.messaging.agent_message.AgentMessage
```

Class representing a request to perform a menu action.

```
class Meta  
    Bases: object  
    Perform metadata.  
  
    handler_class = 'aries_cloudagent.protocols.actionmenu.handlers.perform_handler.PerformHandler'  
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/action-menu/1.0/perform'  
    schema_class = 'PerformSchema'
```

```
class aries_cloudagent.protocols.actionmenu.messages.perform.PerformSchema(*args,  
                                                                    **kwargs)  
  
Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
```

Perform schema class.

```
class Meta  
    Bases: object  
    Perform schema metadata.  
  
    model_class  
        alias of Perform  
  
    name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re  
    params = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None, re
```

aries_cloudagent.protocols.actionmenu.models package

Submodules

aries_cloudagent.protocols.actionmenu.models.menu_form module

Record used to represent the form associated with an action menu option.

```

class aries_cloudagent.protocols.actionmenu.models.menu_form.MenuForm(*,
                                                                    title:
                                                                    str =
                                                                    None,
                                                                    de-
                                                                    scrip-
                                                                    tion:
                                                                    str =
                                                                    None,
                                                                    params:
                                                                    Se-
                                                                    quence[aries_cloudagent.protocols.actionmenu.models.menu_form.MenuFormParam] =
                                                                    None,
                                                                    sub-
                                                                    mit_label:
                                                                    str =
                                                                    None)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Instance of a form associated with an action menu item.

```
class Meta
```

Bases: *object*

Menu form metadata.

```
    schema_class = 'MenuFormSchema'
```

```

class aries_cloudagent.protocols.actionmenu.models.menu_form.MenuFormSchema(*args,
                                                                              **kwargs)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

MenuForm schema.

```
class Meta
```

Bases: *object*

MenuFormSchema metadata.

```
    model_class
```

alias of *MenuForm*

```
    description = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
```

```
    params = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, required=False)>
```

```
    submit_label = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
```

```
    title = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, required=False)>
```

aries_cloudagent.protocols.actionmenu.models.menu_form_param module

Record used to represent a parameter in a menu form.

```

class aries_cloudagent.protocols.actionmenu.models.menu_form_param.MenuFormParam(*,
                                                                                    name:
                                                                                    str
                                                                                    =
                                                                                    None,
                                                                                    ti-
                                                                                    tle:
                                                                                    str
                                                                                    =
                                                                                    None,
                                                                                    de-
                                                                                    fault:
                                                                                    str
                                                                                    =
                                                                                    None,
                                                                                    de-
                                                                                    scrip-
                                                                                    tion:
                                                                                    str
                                                                                    =
                                                                                    None,
                                                                                    in-
                                                                                    put_type:
                                                                                    str
                                                                                    =
                                                                                    None,
                                                                                    re-
                                                                                    quired:
                                                                                    bool
                                                                                    =
                                                                                    None)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Instance of a menu form param associated with an action menu option.

```

class Meta
    Bases: object

    Menu form param metadata.

    schema_class = 'MenuFormParamSchema'

```

```

class aries_cloudagent.protocols.actionmenu.models.menu_form_param.MenuFormParamSchema(*arg, **kw
                                                                                           **kw

```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

MenuFormParam schema.

```

class Meta
    Bases: object

    MenuFormParamSchema metadata.

    model_class
        alias of MenuFormParam

```

```

default = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,

```

```

description = <fields.String(default=<marshmallow.missing>, attribute=None, validate=N

```

```

input_type = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re
required = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None
title = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r

```

aries_cloudagent.protocols.actionmenu.models.menu_option module

Record used to represent individual menu options in an action menu.

```

class aries_cloudagent.protocols.actionmenu.models.menu_option.MenuOption(*,
                                                                           name:
                                                                           str
                                                                           =
                                                                           None,
                                                                           ti-
                                                                           tle:
                                                                           str
                                                                           =
                                                                           None,
                                                                           de-
                                                                           scrip-
                                                                           tion:
                                                                           str
                                                                           =
                                                                           None,
                                                                           dis-
                                                                           abled:
                                                                           bool
                                                                           =
                                                                           None,
                                                                           form:
                                                                           aries_cloudagent.protoc
                                                                           =
                                                                           None)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Instance of a menu option associated with an action menu.

```

class Meta
    Bases: object
    Menu option metadata.
    schema_class = 'MenuOptionSchema'

```

```

class aries_cloudagent.protocols.actionmenu.models.menu_option.MenuOptionSchema(*args,
                                                                           **kwargs)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

MenuOption schema.

```

class Meta
    Bases: object
    MenuOptionSchema metadata.

```

```
model_class
    alias of MenuOption

description = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
disabled = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None)
form = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, re
name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re
title = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
```

Submodules

aries_cloudagent.protocols.actionmenu.base_service module

Base action menu service classes.

```
class aries_cloudagent.protocols.actionmenu.base_service.BaseMenuService (context:
                                                                    aries_cloudagent.config.in
```

Bases: `abc.ABC`

Base action menu service interface.

```
get_active_menu (connection: aries_cloudagent.connections.models.connection_record.ConnectionRecord
                  = None, thread_id: str = None) → aries_cloudagent.protocols.actionmenu.messages.menu.Menu
    Render the current menu.
```

Parameters

- **connection** – The active connection record
- **thread_id** – The thread identifier from the requesting message.

```
perform_menu_action (action_name: str, action_params: dict, connection:
                      aries_cloudagent.connections.models.connection_record.ConnectionRecord
                      = None, thread_id: str = None) →
                      aries_cloudagent.messaging.agent_message.AgentMessage
    Perform an action defined by the active menu.
```

Parameters

- **action_name** – The unique name of the action being performed
- **action_params** – A collection of parameters for the action
- **connection** – The active connection record
- **thread_id** – The thread identifier from the requesting message.

```
classmethod service_handler ()
    Quick accessor for conductor to use.
```

aries_cloudagent.protocols.actionmenu.controller module

Protocol controller for the action menu message family.

```
class aries_cloudagent.protocols.actionmenu.controller.Controller (protocol:
                                                                    str)
```

Bases: `object`

Action menu protocol controller.

determine_roles (*context: aries_cloudagent.config.injection_context.InjectionContext*) → Sequence[str]
 Determine what action menu roles are defined.

aries_cloudagent.protocols.actionmenu.driver_service module

Driver-based action menu service classes.

class aries_cloudagent.protocols.actionmenu.driver_service.**DriverMenuService** (*context: aries_cloudagent.config.injection_context.InjectionContext*)
 Bases: *aries_cloudagent.protocols.actionmenu.base_service.BaseMenuService*

Driver-based action menu service.

get_active_menu (*connection: aries_cloudagent.connections.models.connection_record.ConnectionRecord*
= None, thread_id: str = None) → aries_cloudagent.protocols.actionmenu.messages.menu.Menu
 Render the current menu.

Parameters

- **connection** – The active connection record
- **thread_id** – The thread identifier from the requesting message.

perform_menu_action (*action_name: str, action_params: dict, connection: aries_cloudagent.connections.models.connection_record.ConnectionRecord*
= None, thread_id: str = None) → aries_cloudagent.messaging.agent_message.AgentMessage
 Perform an action defined by the active menu.

Parameters

- **action_name** – The unique name of the action being performed
- **action_params** – A collection of parameters for the action
- **connection** – The active connection record
- **thread_id** – The thread identifier from the requesting message.

send_webhook (*topic: str, payload: dict*)
 Dispatch a webhook through the registered responder.

aries_cloudagent.protocols.actionmenu.message_types module

Message type identifiers for Action Menus.

aries_cloudagent.protocols.actionmenu.routes module

Action menu admin routes.

```
class aries_cloudagent.protocols.actionmenu.routes.MenuJsonSchema (*,
                                                                    only=None,
                                                                    exclude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Matches MenuSchema but without the inherited AgentMessage properties.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.actionmenu.routes.PerformRequestSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Request schema for performing a menu action.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.actionmenu.routes.SendMenuSchema (*,
                                                                    only=None,
                                                                    exclude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Request schema for sending a menu to a connection.

opts = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.protocols.actionmenu.routes.actionmenu_close(request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object at
                                                                0x7f9e74fbd898>)
```


Request handler for closing the menu associated with a connection.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.actionmenu.routes.actionmenu_fetch(request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object at
                                                                0x7f9e74fbd898>)
```

Request handler for fetching the previously-received menu for a connection.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.actionmenu.routes.actionmenu_perform(request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object at
                                                                0x7f9e74fbd898>)
```

Request handler for performing a menu action.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.actionmenu.routes.actionmenu_request(request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object at
                                                                0x7f9e74fbd898>)
```

Request handler for requesting a menu from the connection target.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.actionmenu.routes.actionmenu_send(request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object at
                                                                0x7f9e74fbd898>)
```

Request handler for requesting a menu from the connection target.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.actionmenu.routes.register(app:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object at 0x7f9e74fbd898>)
```

Register routes.

aries_cloudagent.protocols.actionmenu.util module

Action menu utility methods.

```
aries_cloudagent.protocols.actionmenu.util.retrieve_connection_menu(connection_id:
                                                                str,
                                                                context:
                                                                aries_cloudagent.config.injection_context
                                                                →
                                                                aries_cloudagent.protocols.actionmenu.ActionMenu)
```

Retrieve the previously-received action menu.

```
aries_cloudagent.protocols.actionmenu.util.save_connection_menu(menu:
                                                                aries_cloudagent.protocols.actionmenu.ActionMenu,
                                                                connection_id:
                                                                str, context:
                                                                aries_cloudagent.config.injection_context)
```

Save a received action menu.

3.1.2 aries_cloudagent.protocols.basicmessage package

Subpackages

[aries_cloudagent.protocols.basicmessage.handlers package](#)

Submodules

[aries_cloudagent.protocols.basicmessage.handlers.basicmessage_handler module](#)

Basic message handler.

```
class aries_cloudagent.protocols.basicmessage.handlers.basicmessage_handler.BasicMessageHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for basic messages.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for basic messages.
```

Parameters

- **context** – request context
- **responder** – responder callback

[aries_cloudagent.protocols.basicmessage.messages package](#)

Submodules

[aries_cloudagent.protocols.basicmessage.messages.basicmessage module](#)

Basic message.

```
class aries_cloudagent.protocols.basicmessage.messages.basicmessage.BasicMessage (*,
    sent_time:
        Union[str,
        date-
        time.datetime,
        =
        None,
        con-
        tent:
        str
        =
        None,
        lo-
        cal-
        iza-
        tion:
        str
        =
        None,
        **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class defining the structure of a basic message.

class Meta

Bases: *object*

Basic message metadata class.

handler_class = 'aries_cloudagent.protocols.basicmessage.handlers.basicmessage_han

message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/basicmessage/1.0/message'

schema_class = 'BasicMessageSchema'

class aries_cloudagent.protocols.basicmessage.messages.basicmessage.**BasicMessageSchema** (*arg, **kw

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Basic message schema class.

class Meta

Bases: *object*

Basic message schema metadata.

model_class

alias of *BasicMessage*

content = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,

sent_time = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar

Submodules

aries_cloudagent.protocols.basicmessage.message_types module

Message type identifiers for Connections.

aries_cloudagent.protocols.basicmessage.routes module

Basic message admin routes.

class aries_cloudagent.protocols.basicmessage.routes.**SendMessageSchema** (*, only=None, ex-clude=(), many=False, con-text=None, load_only=(), dump_only=(), par-tial=False, un-known=None)

Bases: *marshmallow.schema.Schema*

Request schema for sending a message.

opts = <marshmallow.schema.SchemaOpts object>

```
aries_cloudagent.protocols.basicmessage.routes.connections_send_message(request:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object
                                                                    at
                                                                    0x7f9e74f36048>)
```

Request handler for sending a basic message to a connection.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.basicmessage.routes.register(app:
                                                                    <sphinx.ext.autodoc.importer._MockObject
                                                                    object
                                                                    at
                                                                    0x7f9e74f36048>)
```

Register routes.

3.1.3 aries_cloudagent.protocols.connections package

Subpackages

[aries_cloudagent.protocols.connections.handlers package](#)

Submodules

[aries_cloudagent.protocols.connections.handlers.connection_invitation_handler module](#)

Connect invitation handler.

```
class aries_cloudagent.protocols.connections.handlers.connection_invitation_handler.Connect
Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Handler class for connection invitations.

```
handle (context: aries_cloudagent.messaging.request_context.RequestContext, responder:
aries_cloudagent.messaging.responder.BaseResponder)
Handle connection invitation.
```

Parameters

- **context** – Request context
- **responder** – Responder callback

[aries_cloudagent.protocols.connections.handlers.connection_request_handler module](#)

Connection request handler.

```
class aries_cloudagent.protocols.connections.handlers.connection_request_handler.Connection
Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Handler class for connection requests.

```
handle (context: aries_cloudagent.messaging.request_context.RequestContext, responder:
aries_cloudagent.messaging.responder.BaseResponder)
Handle connection request.
```

Parameters

- **context** – Request context
- **responder** – Responder callback

aries_cloudagent.protocols.connections.handlers.connection_response_handler module

Connection response handler.

class `aries_cloudagent.protocols.connections.handlers.connection_response_handler.ConnectionResponseHandler`
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler class for connection responses.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Handle connection response.

Parameters

- **context** – Request context
- **responder** – Responder callback

aries_cloudagent.protocols.connections.messages package

Submodules

aries_cloudagent.protocols.connections.messages.connection_invitation module

Represents an invitation message for establishing connection.

```
class aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a connection invitation.

```
class Meta
```

Bases: *object*

Metadata for a connection invitation.

```
    handler_class = 'aries_cloudagent.protocols.connections.handlers.connection_invitation_handler.ConnectionInvitationHandler'
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/connections/1.0/invitation'
```

```
    schema_class = 'ConnectionInvitationSchema'
```

```
classmethod from_url(url: str) → aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation
```

Parse a URL-encoded invitation into a *ConnectionInvitation* message.

Parameters url – Url to decode

Returns A *ConnectionInvitation* object.

```
to_url(base_url: str = None) → str
```

Convert an invitation to URL format for sharing.

Returns An invite url

class aries_cloudagent.protocols.connections.messages.connection_invitation.**ConnectionInvitation**

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Connection invitation schema class.

class **Meta**

Bases: *object*

Connection invitation schema metadata.

model_class

alias of *ConnectionInvitation*

did = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitationSchema.validate_data>)>

endpoint = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>

image_url = <fields.Url(default=<marshmallow.missing>, attribute=None, validate=None, if_missing=None)>

label = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_missing=None)>

recipient_keys = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, if_missing=None)>

routing_keys = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, if_missing=None)>

validate_fields(data, **kwargs)

Validate schema fields.

Parameters **data** – The data to validate

Raises *ValidationError* – If any of the fields do not validate

aries_cloudagent.protocols.connections.messages.connection_request module

Represents a connection request message.

class aries_cloudagent.protocols.connections.messages.connection_request.**ConnectionRequest**

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a connection request.

```
class Meta
    Bases: object
    Metadata for a connection request.
    handler_class = 'aries_cloudagent.protocols.connections.handlers.connection_request'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/connections/1.0/request'
    schema_class = 'ConnectionRequestSchema'
```

```
class aries_cloudagent.protocols.connections.messages.connection_request.ConnectionRequestSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Connection request schema class.

```
class Meta
    Bases: object
    Connection request schema metadata.
    model_class
        alias of ConnectionRequest
    connection = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None)>
    image_url = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    label = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, required=True)>
```

aries_cloudagent.protocols.connections.messages.connection_response module

Represents a connection response message.

```
class aries_cloudagent.protocols.connections.messages.connection_response.ConnectionResponseSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a connection response.

```
class Meta
    Bases: object
    Metadata for a connection response.
    handler_class = 'aries_cloudagent.protocols.connections.handlers.connection_response'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/connections/1.0/response'
    schema_class = 'ConnectionResponseSchema'
```

```
class aries_cloudagent.protocols.connections.messages.connection_response.ConnectionResponseSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Connection response schema class.


```

class Meta
    Bases: object

    Connection response schema metadata.

    model_class
        alias of ConnectionResponse

    signed_fields = ('connection',)

    connection = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None)

```

aries_cloudagent.protocols.connections.messages.problem_report module

Represents a connection problem report message.

```

class aries_cloudagent.protocols.connections.messages.problem_report.ProblemReport(*,
                                                                                     prob-
                                                                                     lem_code:
                                                                                     str
                                                                                     =
                                                                                     None,
                                                                                     ex-
                                                                                     plain:
                                                                                     str
                                                                                     =
                                                                                     None,
                                                                                     **kwargs)

```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Base class representing a connection problem report message.

```

class Meta
    Bases: object

    Connection problem report metadata.

    handler_class = 'aries_cloudagent.messaging.problem_report.handler.ProblemReportHandler'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/connections/1.0/problem_report'
    schema_class = 'ProblemReportSchema'

```

```

class aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportReason
    Bases: str, enum.Enum

    Supported reason codes.

    INVITATION_NOT_ACCEPTED = 'invitation_not_accepted'
    REQUEST_NOT_ACCEPTED = 'request_not_accepted'
    REQUEST_PROCESSING_ERROR = 'request_processing_error'
    RESPONSE_NOT_ACCEPTED = 'response_not_accepted'
    RESPONSE_PROCESSING_ERROR = 'response_processing_error'

```

```

class aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportSchema(*,
                                                                                         **kwargs)
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema

    Schema for ProblemReport base class.

```

```
class Meta
    Bases: object
    Metadata for problem report schema.
    model_class
        alias of ProblemReport
    explain = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    problem_code = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
```

aries_cloudagent.protocols.connections.models package

Submodules

aries_cloudagent.protocols.connections.models.connection_detail module

An object for containing the connection request/response DID information.

```
class aries_cloudagent.protocols.connections.models.connection_detail.ConnectionDetail(*,
    did: str
    = None
    did_c
    aries
    =
    None
    **kw
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing the details of a connection.

```
class Meta
    Bases: object
    ConnectionDetail metadata.
    schema_class = 'ConnectionDetailSchema'
```

did
Accessor for the connection DID.

Returns The DID for this connection

did_doc
Accessor for the connection DID Document.

Returns The DIDDoc for this connection

```
class aries_cloudagent.protocols.connections.models.connection_detail.ConnectionDetailSchema
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

ConnectionDetail schema.

```
class Meta
    Bases: object
    ConnectionDetailSchema metadata.
```

```

    model_class = 'ConnectionDetail'

    did = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<aries_cl
    did_doc = <fields.DIDDocWrapper(default=<marshmallow.missing>, attribute=None, validate=
class aries_cloudagent.protocols.connections.models.connection_detail.DIDDocWrapper(*,
    de-
    fault=<m
    miss-
    ing=<ma
    data_key=
    at-
    tribute=N
    val-
    i-
    date=Nor
    re-
    quired=F
    al-
    low_none
    load_only
    dump_onl
    er-
    ror_messa
    **meta-
    data)

```

Bases: `marshmallow.fields.Field`

Field that loads and serializes DIDDoc.

Submodules

aries_cloudagent.protocols.connections.manager module

Classes to manage connections.

```

class aries_cloudagent.protocols.connections.manager.ConnectionManager(context:
    aries_cloudagent.config.injector.Injector)

```

Bases: `object`

Class for managing connections.

```

RECORD_TYPE_DID_DOC = 'did_doc'

```

```

RECORD_TYPE_DID_KEY = 'did_key'

```

```

accept_response(response: aries_cloudagent.protocols.connections.messages.connection_response.ConnectionResponse,
    receipt: aries_cloudagent.transport.inbound.receipt.MessageReceipt) →
    aries_cloudagent.connections.models.connection_record.ConnectionRecord

```

Accept a connection response.

Process a ConnectionResponse message by looking up the connection request and setting up the pairwise connection.

Parameters

- **response** – The *ConnectionResponse* to accept
- **receipt** – The message receipt

Returns The updated *ConnectionRecord* representing the connection

Raises

- *ConnectionManagerError* – If there is no DID associated with the connection response
- *ConnectionManagerError* – If the corresponding connection is not at the request or response stage

add_key_for_did (*did: str, key: str*)

Store a verkey for lookup against a DID.

Parameters

- **did** – The DID to associate with this key
- **key** – The verkey to be added

context

Accessor for the current injection context.

Returns The injection context for this connection manager

create_did_document (*did_info: aries_cloudagent.wallet.base.DIDInfo, inbound_connection_id: str = None, svc_endpoints: Sequence[str] = []*) → *aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc*

Create our DID document for a given DID.

Parameters

- **did_info** – The DID information (DID and verkey) used in the connection
- **inbound_connection_id** – The ID of the inbound routing connection to use
- **svc_endpoints** – Custom endpoints for the DID Document

Returns The prepared *DIDDoc* instance

create_invitation (*my_label: str = None, my_endpoint: str = None, their_role: str = None, accept: str = None, public: bool = False, multi_use: bool = False, alias: str = None*) → *Tuple[aries_cloudagent.connections.models.connection_record.ConnectionRecord, aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation]*

Generate new connection invitation.

This interaction represents an out-of-band communication channel. In the future and in practice, these sort of invitations will be received over any number of channels such as SMS, Email, QR Code, NFC, etc.

Structure of an invite message:

```
{
  "@type":
    "did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/connections/1.0/invitation",
  "label": "Alice",
  "did": "did:sov:QmWbsNYhMrjHiqZDTUTEJs"
}
```

Or, in the case of a peer DID:

```
{
  "@type":
    "did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/connections/1.0/invitation",
  "label": "Alice",

```

(continues on next page)

(continued from previous page)

```

"did": "did:peer:oiSqsNYhMrjHiqZDTUthsw",
"recipientKeys": ["8HH5gYEeNc3z7PYXmd54d4x6qAfCNrqQqEB3nS7Zfu7K"],
"serviceEndpoint": "https://example.com/endpoint"
}

```

Currently, only peer DID is supported.

Parameters

- **my_label** – label for this connection
- **my_endpoint** – endpoint where other party can reach me
- **their_role** – a role to assign the connection
- **accept** – set to 'auto' to auto-accept a corresponding connection request
- **public** – set to True to create an invitation from the public DID
- **multi_use** – set to True to create an invitation for multiple use
- **alias** – optional alias to apply to connection for later use

Returns A tuple of the new *ConnectionRecord* and *ConnectionInvitation* instances

create_request (*connection*: *aries_cloudagent.connections.models.connection_record.ConnectionRecord*,
my_label: *str* = *None*, *my_endpoint*: *str* = *None*) →
aries_cloudagent.protocols.connections.messages.connection_request.ConnectionRequest

Create a new connection request for a previously-received invitation.

Parameters

- **connection** – The *ConnectionRecord* representing the invitation to accept
- **my_label** – My label
- **my_endpoint** – My endpoint

Returns A new *ConnectionRequest* message to send to the other agent

create_response (*connection*: *aries_cloudagent.connections.models.connection_record.ConnectionRecord*,
my_endpoint: *str* = *None*) → *aries_cloudagent.protocols.connections.messages.connection_response.ConnectionResponse*

Create a connection response for a received connection request.

Parameters

- **connection** – The *ConnectionRecord* with a pending connection request
- **my_endpoint** – The endpoint I can be reached at

Returns A tuple of the updated *ConnectionRecord* new *ConnectionResponse* message

create_static_connection (*my_did*: *str* = *None*, *my_seed*: *str* = *None*, *their_did*: *str* = *None*,
their_seed: *str* = *None*, *their_verkey*: *str* = *None*, *their_endpoint*:
str = *None*, *their_role*: *str* = *None*, *their_label*: *str* = *None*, *alias*:
str = *None*) → (<class 'aries_cloudagent.wallet.base.DIDInfo'>,
<class 'aries_cloudagent.wallet.base.DIDInfo'>, <class
'aries_cloudagent.connections.models.connection_record.ConnectionRecord'>)

Register a new static connection (for use by the test suite).

Parameters

- **my_did** – override the DID used in the connection
- **my_seed** – provide a seed used to generate our DID and keys

- **their_did** – provide the DID used by the other party
- **their_seed** – provide a seed used to generate their DID and keys
- **their_verkey** – provide the verkey used by the other party
- **their_endpoint** – their URL endpoint for routing messages
- **their_role** – their role in this connection
- **alias** – an alias for this connection record

Returns The new *ConnectionRecord* instance

diddoc_connection_targets (*doc: aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc*,
sender_verkey: str, their_label: str = None) → Sequence[aries_cloudagent.connections.models.connection_target.ConnectionTarget]
 Get a list of connection targets from a DID Document.

Parameters

- **doc** – The DID Document to create the target from
- **sender_verkey** – The verkey we are using
- **their_label** – The connection label they are using

establish_inbound (*connection: aries_cloudagent.connections.models.connection_record.ConnectionRecord*,
inbound_connection_id: str, outbound_handler) → str
 Assign the inbound routing connection for a connection record.

Returns: the current routing state (request or done)

fetch_connection_targets (*connection: aries_cloudagent.connections.models.connection_record.ConnectionRecord*)
 → Sequence[aries_cloudagent.connections.models.connection_target.ConnectionTarget]
 Get a list of connection target from a *ConnectionRecord*.

Parameters connection – The connection record (with associated *DIDDoc*) used to generate the connection target

fetch_did_document (*did: str*) → Tuple[aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc,
 aries_cloudagent.storage.record.StorageRecord]
 Retrieve a DID Document for a given DID.

Parameters did – The DID to search for

find_connection (*their_did: str, my_did: str = None, my_verkey: str = None*,
auto_complete=False) → aries_cloudagent.connections.models.connection_record.ConnectionRecord
 Look up existing connection information for a sender verkey.

Parameters

- **their_did** – Their DID
- **my_did** – My DID
- **my_verkey** – My verkey
- **auto_complete** – Should this connection automatically be promoted to active

Returns The located *ConnectionRecord*, if any

find_did_for_key (*key: str*) → str
 Find the DID previously associated with a key.

Parameters key – The verkey to look up

find_inbound_connection (*receipt: aries_cloudagent.transport.inbound.receipt.MessageReceipt*)
 → *aries_cloudagent.connections.models.connection_record.ConnectionRecord*
 Deserialize an incoming message and further populate the request context.

Parameters **receipt** – The message receipt

Returns The *ConnectionRecord* associated with the expanded message, if any

get_connection_targets (*, *connection_id: str = None, connection: aries_cloudagent.connections.models.connection_record.ConnectionRecord = None*)
 Create a connection target from a *ConnectionRecord*.

Parameters

- **connection_id** – The connection ID to search for
- **connection** – The connection record itself, if already available

receive_invitation (*invitation: aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation*,
 their_role: str = None, accept: str = None, alias: str = None) →
 aries_cloudagent.connections.models.connection_record.ConnectionRecord
 Create a new connection record to track a received invitation.

Parameters

- **invitation** – The *ConnectionInvitation* to store
- **their_role** – The role assigned to this connection
- **accept** – set to 'auto' to auto-accept the invitation
- **alias** – optional alias to set on the record

Returns The new *ConnectionRecord* instance

receive_request (*request: aries_cloudagent.protocols.connections.messages.connection_request.ConnectionRequest*,
 receipt: aries_cloudagent.transport.inbound.receipt.MessageReceipt) →
 aries_cloudagent.connections.models.connection_record.ConnectionRecord
 Receive and store a connection request.

Parameters

- **request** – The *ConnectionRequest* to accept
- **receipt** – The message receipt

Returns The new or updated *ConnectionRecord* instance

remove_keys_for_did (*did: str*)
 Remove all keys associated with a DID.

Parameters **did** – The DID to remove keys for

resolve_inbound_connection (*receipt: aries_cloudagent.transport.inbound.receipt.MessageReceipt*)
 → *aries_cloudagent.connections.models.connection_record.ConnectionRecord*
 Populate the receipt DID information and find the related *ConnectionRecord*.

Parameters **receipt** – The message receipt

Returns The *ConnectionRecord* associated with the expanded message, if any

store_did_document (*did_doc: aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc*)
 Store a DID document.

Parameters **did_doc** – The *DIDDoc* instance to be persisted

update_inbound (*inbound_connection_id: str, recip_verkey: str, routing_state: str*)

Activate connections once a route has been established.

Looks up pending connections associated with the inbound routing connection and marks the routing as complete.

exception `aries_cloudagent.protocols.connections.manager.ConnectionManagerError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Connection error.

`aries_cloudagent.protocols.connections.message_types` module

Message type identifiers for Connections.

`aries_cloudagent.protocols.connections.routes` module

Connection handling admin routes.

class `aries_cloudagent.protocols.connections.routes.ConnectionListSchema` (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)

Bases: `marshmallow.schema.Schema`

Result schema for connection list.

opts = `<marshmallow.schema.SchemaOpts object>`

class `aries_cloudagent.protocols.connections.routes.ConnectionStaticRequestSchema` (*, only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)

Bases: `marshmallow.schema.Schema`

Request schema for a new static connection.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.connections.routes.ConnectionStaticResultSchema(*,
    only=None,
    ex-
    clude=(),
    many=False,
    con-
    text=None,
    load_only=(),
    dump_only=(),
    par-
    tial=False,
    un-
    known=None)
```

Bases: `marshmallow.schema.Schema`

Result schema for new static connection.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.connections.routes.InvitationResultSchema(*,
    only=None,
    ex-
    clude=(),
    many=False,
    con-
    text=None,
    load_only=(),
    dump_only=(),
    par-
    tial=False,
    un-
    known=None)
```

Bases: `marshmallow.schema.Schema`

Result schema for a new connection invitation.

opts = `<marshmallow.schema.SchemaOpts object>`

`aries_cloudagent.protocols.connections.routes.connection_sort_key(conn)`

Get the sorting key for a particular connection.

```
aries_cloudagent.protocols.connections.routes.connections_accept_invitation(request:
    <sphinx.ext.autodoc.t
    ob-
    ject
    at
    0x7f9e751a5b00>)
```

Request handler for accepting a stored connection invitation.

Parameters **request** – aiohttp request object

Returns The resulting connection record details

```
aries_cloudagent.protocols.connections.routes.connections_accept_request (request:
    <sphinx.ext.autodoc.importer.MockObject object at
    0x7f9e751a5b00>)
```

Request handler for accepting a stored connection request.

Parameters **request** – aiohttp request object

Returns The resulting connection record details

```
aries_cloudagent.protocols.connections.routes.connections_create_invitation (request:
    <sphinx.ext.autodoc.importer.MockObject object at
    0x7f9e751a5b00>)
```

Request handler for creating a new connection invitation.

Parameters **request** – aiohttp request object

Returns The connection invitation details

```
aries_cloudagent.protocols.connections.routes.connections_create_static (request:
    <sphinx.ext.autodoc.importer.MockObject object at
    0x7f9e751a5b00>)
```

Request handler for creating a new static connection.

Parameters **request** – aiohttp request object

Returns The new connection record

```
aries_cloudagent.protocols.connections.routes.connections_establish_inbound (request:
    <sphinx.ext.autodoc.importer.MockObject object at
    0x7f9e751a5b00>)
```

Request handler for setting the inbound connection on a connection record.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.connections.routes.connections_list (request:
    <sphinx.ext.autodoc.importer.MockObject object at
    0x7f9e751a5b00>)
```

Request handler for searching connection records.

Parameters **request** – aiohttp request object

Returns The connection list response

```
aries_cloudagent.protocols.connections.routes.connections_receive_invitation (request:
    <sphinx.ext.autodoc.importer.MockObject object at
    0x7f9e751a5b00>)
```

Request handler for receiving a new connection invitation.

Parameters **request** – aiohttp request object

Returns The resulting connection record details

```
aries_cloudagent.protocols.connections.routes.connections_remove(request:
                                                                    <sphinx.ext.autodoc.importer._MockC
                                                                    object      at
                                                                    0x7f9e751a5b00>)
```

Request handler for removing a connection record.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.connections.routes.connections_retrieve(request:
                                                                    <sphinx.ext.autodoc.importer._MockC
                                                                    object      at
                                                                    0x7f9e751a5b00>)
```

Request handler for fetching a single connection record.

Parameters **request** – aiohttp request object

Returns The connection record response

```
aries_cloudagent.protocols.connections.routes.register(app:
                                                        <sphinx.ext.autodoc.importer._MockObject
                                                        object      at
                                                        0x7f9e751a5b00>)
```

Register routes.

3.1.4 aries_cloudagent.protocols.credentials package

Subpackages

aries_cloudagent.protocols.credentials.handlers package

Submodules

aries_cloudagent.protocols.credentials.handlers.credential_issue_handler module

Basic message handler.

```
class aries_cloudagent.protocols.credentials.handlers.credential_issue_handler.CredentialIssueHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential offers.

```
handle (context: aries_cloudagent.messaging.request_context.RequestContext, responder:
            aries_cloudagent.messaging.responder.BaseResponder)
    Message handler logic for credential offers.
```

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.credentials.handlers.credential_offer_handler module

Basic message handler.

class aries_cloudagent.protocols.credentials.handlers.credential_offer_handler.CredentialOfferHandler

Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential offers.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
 Message handler logic for credential offers.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.credentials.handlers.credential_request_handler module

Credential request handler.

class aries_cloudagent.protocols.credentials.handlers.credential_request_handler.CredentialRequestHandler

Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential requests.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
 Message handler logic for credential requests.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.credentials.handlers.credential_stored_handler module

Credential stored message handler.

class aries_cloudagent.protocols.credentials.handlers.credential_stored_handler.CredentialStoredHandler

Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential offers.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
 Message handler logic for credential stored messages.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.credentials.messages package

Submodules

aries_cloudagent.protocols.credentials.messages.credential_issue module

A credential content message.

```
class aries_cloudagent.protocols.credentials.messages.credential_issue.CredentialIssue(*,
is-
sue:
str
=
None
**kw)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential.

class Meta

Bases: *object*

Credential metadata.

handler_class = 'aries_cloudagent.protocols.credentials.handlers.credential_issue_h'

message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/credential-issuance/0.1/credential'

schema_class = 'CredentialIssueSchema'

class aries_cloudagent.protocols.credentials.messages.credential_issue.CredentialIssueSchema

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Credential schema.

class Meta

Bases: *object*

Credential schema metadata.

model_class

alias of *CredentialIssue*

issue = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r

aries_cloudagent.protocols.credentials.messages.credential_offer module

A credential offer content message.

```

class aries_cloudagent.protocols.credentials.messages.credential_offer.CredentialOffer(*,
of-
fer_j-
son =
None,
credential_
preview =
None,
comment =
None,
offer_json =
None,
**kw)

```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential offer.

```

class Meta
    Bases: object
    CredentialOffer metadata.
    handler_class = 'aries_cloudagent.protocols.credentials.handlers.credential_offer_h
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/credential-issuance/0.1/credent
    schema_class = 'CredentialOfferSchema'

```

```

class aries_cloudagent.protocols.credentials.messages.credential_offer.CredentialOfferSchema

```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Credential offer schema.

```

class Meta
    Bases: object
    Credential offer schema metadata.
    model_class
        alias of CredentialOffer
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    credential_preview = <fields.Dict(default=<marshmallow.missing>, attribute=None, valid
    offer_json = <fields.String(default=<marshmallow.missing>, attribute=None, validate=No

```

aries_cloudagent.protocols.credentials.messages.credential_request module

A credential request content message.

```
class aries_cloudagent.protocols.credentials.messages.credential_request.CredentialRequest
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential request.

```
class Meta
```

Bases: *object*

CredentialRequest metadata.

```
    handler_class = 'aries_cloudagent.protocols.credentials.handlers.credential_request'
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/credential-issuance/0.1/credential_request'
```

```
    schema_class = 'CredentialRequestSchema'
```

```
class aries_cloudagent.protocols.credentials.messages.credential_request.CredentialRequestSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Credential request schema.

```
class Meta
```

Bases: *object*

Credential request schema metadata.

```
    model_class
```

alias of *CredentialRequest*

```
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)>
```

```
    request = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)>
```

aries_cloudagent.protocols.credentials.messages.credential_stored module

A credential stored message.

```
class aries_cloudagent.protocols.credentials.messages.credential_stored.CredentialStored(*args, **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential stored message.

```
class Meta
```

Bases: *object*

Credential metadata.

```
    handler_class = 'aries_cloudagent.protocols.credentials.handlers.credential_stored'
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/credential-issuance/0.1/credential_stored'
```

```
    schema_class = 'CredentialStoredSchema'
class aries_cloudagent.protocols.credentials.messages.credential_stored.CredentialStoredSchema
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Credential stored schema.
    class Meta
        Bases: object
        Schema metadata.
    model_class
        alias of CredentialStored
```

aries_cloudagent.protocols.credentials.models package

Submodules

aries_cloudagent.protocols.credentials.models.credential_exchange module

Handle credential exchange information interface with non-secrets storage.


```
class aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange
```

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecord*

Represents a credential exchange.

INITIATOR_EXTERNAL = 'external'

INITIATOR_SELF = 'self'

LOG_STATE_FLAG = 'debug.credentials'

class Meta

Bases: *object*

CredentialExchange metadata.

schema_class = 'CredentialExchangeSchema'

RECORD_ID_NAME = 'credential_exchange_id'

RECORD_TYPE = 'credential_exchange'

STATE_CREDENTIAL_RECEIVED = 'credential_received'

STATE_ISSUED = 'issued'

STATE_OFFER_RECEIVED = 'offer_received'

STATE_OFFER_SENT = 'offer_sent'

STATE_REQUEST_RECEIVED = 'request_received'

STATE_REQUEST_SENT = 'request_sent'

STATE_STORED = 'stored'

TAG_NAMES = {'thread_id'}

WEBHOOK_TOPIC = 'credentials'

credential_exchange_id

Accessor for the ID associated with this exchange.

post_save (*context: aries_cloudagent.config.injection_context.InjectionContext, new_record: bool, last_state: str, webhook: bool = None*)

Perform post-save actions.

Parameters

- **context** – The injection context to use
- **new_record** – Flag indicating if the record was just created
- **last_state** – The previous state value
- **webhook** – Adjust whether the webhook is called

record_value

Accessor to for the JSON record value props for this credential exchange.

classmethod retrieve_by_thread_and_initiator (*context:*

aries_cloudagent.config.injection_context.InjectionContext,
thread_id: str, initiator: str) →
aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange

Retrieve a credential exchange record by thread ID and initiator.

class *aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange*

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecordSchema*

Schema to allow serialization/deserialization of credential exchange records.

```

class Meta
    Bases: object

    CredentialExchangeSchema metadata.

    model_class
        alias of CredentialExchange

    auto_issue = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None)
    connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    credential = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_definition_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_exchange_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_offer = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_request = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_request_metadata = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
    credential_values = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
    error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    parent_thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    raw_credential = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
    schema_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
    state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, required=True)
    thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)

```

Submodules

aries_cloudagent.protocols.credentials.manager module

Classes to manage credentials.

```

class aries_cloudagent.protocols.credentials.manager.CredentialManager(context:
                                                                    aries_cloudagent.config.injector.Injector)

```

Bases: `object`

Class for managing credentials.

context

Accessor for the current injection context.

Returns The injection context for this credential manager

```

create_offer(credential_definition_id: str, connection_id: str, auto_issue: bool = None, credential_values: dict = None)

```

Create a new credential exchange representing an offer.

Parameters

- **credential_definition_id** – Credential definition id for offer
- **connection_id** – Connection to create offer for

Returns A new credential exchange record

create_request (*credential_exchange_record: aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeRecord*,
connection_record: aries_cloudagent.connections.models.connection_record.ConnectionRecord)
 Create a credential request.

Parameters

- **credential_exchange_record** – Credential exchange to create request for
- **connection_record** – Connection to create the request for

Returns A tuple (credential_exchange_record, credential_request_message)

credential_stored (*credential_stored_message: aries_cloudagent.protocols.credentials.messages.credential_stored.CredentialStoredMessage*)
 Receive confirmation that holder stored credential.

Parameters **credential_message** – credential to store

issue_credential (*credential_exchange_record: aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeRecord*)
 Issue a credential.

Parameters **credential_exchange_record** – The credential exchange we are issuing a credential for

Returns (Updated credential exchange record, credential message obj)

Return type Tuple

receive_credential (*credential_message: aries_cloudagent.protocols.credentials.messages.credential_issue.CredentialIssueMessage*)
 Receive a credential a credential from an issuer.

Hold in storage to be potentially processed by controller before storing.

Parameters **credential_message** – credential to store

receive_offer (*credential_offer_message: aries_cloudagent.protocols.credentials.messages.credential_offer.CredentialOfferMessage*,
connection_id: str)
 Receive a credential offer.

Parameters

- **credential_offer** – Credential offer to receive
- **connection_id** – Connection to receive offer on

Returns The credential_exchange_record

receive_request (*credential_request_message: aries_cloudagent.protocols.credentials.messages.credential_request.CredentialRequestMessage*)
 Receive a credential request.

Parameters **credential_request_message** – Credential request to receive

store_credential (*credential_exchange_record: aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeRecord*,
credential_id: str = None)
 Store a credential in the wallet.

Parameters

- **credential_message** – credential to store
- **credential_id** – string to use as id for record in wallet

```
exception aries_cloudagent.protocols.credentials.manager.CredentialManagerError(*args,
er-
ror_code:
str
=
None,
**kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Credential error.

aries_cloudagent.protocols.credentials.message_types module

Message type identifiers for credentials.

aries_cloudagent.protocols.credentials.routes module

Credential handling admin routes.

```
class aries_cloudagent.protocols.credentials.routes.CredentialExchangeListSchema(*,
only=None,
ex-
clude=(),
many=False,
con-
text=None,
load_only=(),
dump_only=(),
par-
tial=False,
un-
known=None)
```

Bases: `marshmallow.schema.Schema`

Result schema for a credential exchange query.

`opts = <marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.credentials.routes.CredentialIssueRequestSchema(*,
only=None,
ex-
clude=(),
many=False,
con-
text=None,
load_only=(),
dump_only=(),
par-
tial=False,
un-
known=None)
```

Bases: `marshmallow.schema.Schema`

Request schema for sending a credential issue admin message.

`opts = <marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.credentials.routes.CredentialIssueResultSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Result schema for sending a credential issue admin message.

    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.protocols.credentials.routes.CredentialListSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Result schema for a credential query.

    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.protocols.credentials.routes.CredentialOfferRequestSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Request schema for sending a credential offer admin message.

    opts = <marshmallow.schema.SchemaOpts object>
```

```

class aries_cloudagent.protocols.credentials.routes.CredentialOfferResultSchema(*,
    only=None,
    ex-
    clude=(),
    many=False,
    con-
    text=None,
    load_only=(),
    dump_only=(),
    par-
    tial=False,
    un-
    known=None)

Bases: marshmallow.schema.Schema

Result schema for sending a credential offer admin message.

opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.credentials.routes.CredentialProblemReportRequestSchema(*,
    on-
    ex-
    cl-
    m-
    co-
    te-
    lo-
    du-
    pa-
    tic-
    un-
    kn-

Bases: marshmallow.schema.Schema

Request schema for sending a problem report.

opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.credentials.routes.CredentialRequestResultSchema(*,
    only=None,
    ex-
    clude=(),
    many=False,
    con-
    text=None,
    load_only=(),
    dump_only=(),
    par-
    tial=False,
    un-
    known=None)

Bases: marshmallow.schema.Schema

Result schema for sending a credential request admin message.

opts = <marshmallow.schema.SchemaOpts object>

```

```
class aries_cloudagent.protocols.credentials.routes.CredentialSchema(*  
    only=None,  
    ex-  
    clude=(),  
    many=False,  
    con-  
    text=None,  
    load_only=(),  
    dump_only=(),  
    par-  
    tial=False,  
    un-  
    known=None)  
  
    Bases: marshmallow.schema.Schema  
    Result schema for a credential query.  
  
    opts = <marshmallow.schema.SchemaOpts object>  
  
class aries_cloudagent.protocols.credentials.routes.CredentialSendRequestSchema(*  
    only=None,  
    ex-  
    clude=(),  
    many=False,  
    con-  
    text=None,  
    load_only=(),  
    dump_only=(),  
    par-  
    tial=False,  
    un-  
    known=None)  
  
    Bases: marshmallow.schema.Schema  
    Request schema for sending a credential offer admin message.  
  
    opts = <marshmallow.schema.SchemaOpts object>  
  
class aries_cloudagent.protocols.credentials.routes.CredentialSendResultSchema(*  
    only=None,  
    ex-  
    clude=(),  
    many=False,  
    con-  
    text=None,  
    load_only=(),  
    dump_only=(),  
    par-  
    tial=False,  
    un-  
    known=None)  
  
    Bases: marshmallow.schema.Schema  
    Result schema for sending a credential offer admin message.  
  
    opts = <marshmallow.schema.SchemaOpts object>
```



```
class aries_cloudagent.protocols.credentials.routes.CredentialStoreRequestSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Request schema for sending a credential store admin message.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.credentials.routes.RawEncCredAttrSchema (*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Credential attribute schema.

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.credentials.routes.RevRegSchema (*,
                                                                    only=None,
                                                                    exclude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)

    Bases: marshmallow.schema.Schema

    Revocation registry schema.

    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.protocols.credentials.routes.WitnessSchema(*,
                                                                only=None,
                                                                exclude=(),
                                                                many=False,
                                                                con-
                                                                text=None,
                                                                load_only=(),
                                                                dump_only=(),
                                                                par-
                                                                tial=False,
                                                                un-
                                                                known=None)

Bases: marshmallow.schema.Schema

Witness schema.

opts = <marshmallow.schema.SchemaOpts object>

aries_cloudagent.protocols.credentials.routes.credential_exchange_issue(request:
                                                                <sphinx.ext.autodoc.import
                                                                ob-
                                                                ject
                                                                at
                                                                0x7f9e7520bf28>)

Request handler for sending a credential.

Parameters request – aiohttp request object

Returns The credential details.

aries_cloudagent.protocols.credentials.routes.credential_exchange_list(request:
                                                                <sphinx.ext.autodoc.importe
                                                                ob-
                                                                ject
                                                                at
                                                                0x7f9e7520bf28>)

Request handler for searching credential exchange records.

Parameters request – aiohttp request object

Returns The credential exchange list response

aries_cloudagent.protocols.credentials.routes.credential_exchange_problem_report(request:
                                                                <sphinx.ext.a
                                                                ob-
                                                                ject
                                                                at
                                                                0x7f9e7520bf28>)

Request handler for sending a problem report.

Parameters request – aiohttp request object

aries_cloudagent.protocols.credentials.routes.credential_exchange_remove(request:
                                                                <sphinx.ext.autodoc.impo
                                                                ob-
                                                                ject
                                                                at
                                                                0x7f9e7520bf28>)

Request handler for removing a credential exchange record.

Parameters request – aiohttp request object
```

```
aries_cloudagent.protocols.credentials.routes.credential_exchange_retrieve(request:
    <sphinx.ext.autodoc.importer.Object at 0x7f9e7520bf28>)
```

Request handler for fetching a single credential exchange record.

Parameters **request** – aiohttp request object

Returns The credential exchange record response

```
aries_cloudagent.protocols.credentials.routes.credential_exchange_send(request:
    <sphinx.ext.autodoc.importer.Object at 0x7f9e7520bf28>)
```

Request handler for sending a credential.

Parameters **request** – aiohttp request object

Returns The credential offer details.

```
aries_cloudagent.protocols.credentials.routes.credential_exchange_send_offer(request:
    <sphinx.ext.autodoc.importer.Object at 0x7f9e7520bf28>)
```

Request handler for sending a credential offer.

Parameters **request** – aiohttp request object

Returns The credential offer details.

```
aries_cloudagent.protocols.credentials.routes.credential_exchange_send_request(request:
    <sphinx.ext.autodoc.importer.Object at 0x7f9e7520bf28>)
```

Request handler for sending a credential request.

Parameters **request** – aiohttp request object

Returns The credential request details.

```
aries_cloudagent.protocols.credentials.routes.credential_exchange_store(request:
    <sphinx.ext.autodoc.importer.Object at 0x7f9e7520bf28>)
```

Request handler for storing a credential request.

Parameters **request** – aiohttp request object

Returns The credential request details.

```
aries_cloudagent.protocols.credentials.routes.credentials_get (request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object          at
                                                                0x7f9e7520bf28>)
```

Request handler for retrieving a credential.

Parameters **request** – aiohttp request object

Returns The credential response

```
aries_cloudagent.protocols.credentials.routes.credentials_list (request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object          at
                                                                0x7f9e7520bf28>)
```

Request handler for searching credential records.

Parameters **request** – aiohttp request object

Returns The credential list response

```
aries_cloudagent.protocols.credentials.routes.credentials_remove (request:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object          at
                                                                0x7f9e7520bf28>)
```

Request handler for searching connection records.

Parameters **request** – aiohttp request object

Returns The connection list response

```
aries_cloudagent.protocols.credentials.routes.register (app:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object at 0x7f9e7520bf28>)
```

Register routes.

3.1.5 aries_cloudagent.protocols.discovery package

Subpackages

[aries_cloudagent.protocols.discovery.handlers package](#)

Submodules

[aries_cloudagent.protocols.discovery.handlers.disclose_handler module](#)

Handler for incoming disclose messages.

```
class aries_cloudagent.protocols.discovery.handlers.disclose_handler.DiscloseHandler
Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Handler for incoming disclose messages.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
          aries_cloudagent.messaging.responder.BaseResponder)
Message handler implementation.
```

aries_cloudagent.protocols.discovery.handlers.query_handler module

Handler for incoming query messages.

class aries_cloudagent.protocols.discovery.handlers.query_handler.**QueryHandler**
Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Handler for incoming query messages.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
Message handler implementation.

aries_cloudagent.protocols.discovery.messages package

Submodules

aries_cloudagent.protocols.discovery.messages.disclose module

Represents a feature discovery disclosure message.

class aries_cloudagent.protocols.discovery.messages.disclose.**Disclose** (*, *protocols:* *Sequence[Mapping[str, Mapping[KT, VT_co]]]*, *= None*, ***kwargs*)

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Represents a feature discovery disclosure, the response to a query message.

class **Meta**

Bases: *object*

Disclose metadata.

handler_class = 'aries_cloudagent.protocols.discovery.handlers.disclose_handler.DiscloseHandler'

message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/discover-features/1.0/disclose'

schema_class = 'DiscloseSchema'

class aries_cloudagent.protocols.discovery.messages.disclose.**DiscloseSchema** (*args, ***kwargs*)

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Disclose message schema used in serialization/deserialization.

class **Meta**

Bases: *object*

DiscloseSchema metadata.

model_class

alias of *Disclose*

```
protocols = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None,
class aries_cloudagent.protocols.discovery.messages.disclose.ProtocolDescriptorSchema(*,
only=
ex-
clude=
many=
con-
text=N
load_c
dump_
par-
tial=F
un-
known

Bases: marshmallow.schema.Schema
Schema for an entry in the protocols list.

opts = <marshmallow.schema.SchemaOpts object>
```

aries_cloudagent.protocols.discovery.messages.query module

Represents a feature discovery query message.

```
class aries_cloudagent.protocols.discovery.messages.query.Query(*,      query:
str = None,
comment:
str = None,
**kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Represents a feature discovery query.

Used for inspecting what message types are supported by the agent.

```
class Meta
Bases: object
Query metadata.

handler_class = 'aries_cloudagent.protocols.discovery.handlers.query_handler.QueryH
message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/discover-features/1.0/query'
schema_class = 'QuerySchema'
```

```
class aries_cloudagent.protocols.discovery.messages.query.QuerySchema(*args,
**kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Query message schema used in serialization/deserialization.

```
class Meta
Bases: object
QuerySchema metadata.

model_class
alias of Query

comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

```
query = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r
```

Submodules

aries_cloudagent.protocols.discovery.message_types module

Message type identifiers for Feature Discovery.

aries_cloudagent.protocols.discovery.routes module

Feature discovery admin routes.

```
class aries_cloudagent.protocols.discovery.routes.QueryResultSchema(*,
                                                                    only=None,
                                                                    ex-
                                                                    clude=(),
                                                                    many=False,
                                                                    con-
                                                                    text=None,
                                                                    load_only=(),
                                                                    dump_only=(),
                                                                    par-
                                                                    tial=False,
                                                                    un-
                                                                    known=None)
```

Bases: marshmallow.schema.Schema

Result schema for the protocol list.

opts = <marshmallow.schema.SchemaOpts object>

```
aries_cloudagent.protocols.discovery.routes.query_features(request:
                                                            <sphinx.ext.autodoc.importer._MockObject
                                                            object at
                                                            0x7f9e74e7bb70>)
```

Request handler for inspecting supported protocols.

Parameters **request** – aiohttp request object

Returns The disclosed protocols response

```
aries_cloudagent.protocols.discovery.routes.register(app:
                                                       <sphinx.ext.autodoc.importer._MockObject
                                                       object at 0x7f9e74e7bb70>)
```

Register routes.

3.1.6 aries_cloudagent.protocols.introduction package

Subpackages

aries_cloudagent.protocols.introduction.handlers package

Submodules

`aries_cloudagent.protocols.introduction.handlers.forward_invitation_handler` module

Handler for incoming forward invitation messages.

class `aries_cloudagent.protocols.introduction.handlers.forward_invitation_handler.ForwardInvitationHandler`

Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming forward invitation messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.introduction.handlers.invitation_handler` module

Handler for incoming invitation messages.

class `aries_cloudagent.protocols.introduction.handlers.invitation_handler.InvitationHandler`

Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming invitation messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.introduction.handlers.invitation_request_handler` module

Handler for incoming invitation request messages.

class `aries_cloudagent.protocols.introduction.handlers.invitation_request_handler.InvitationRequestHandler`

Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming invitation request messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.introduction.messages` package

Submodules

`aries_cloudagent.protocols.introduction.messages.forward_invitation` module

Represents a forwarded invitation from another agent.


```
class aries_cloudagent.protocols.introduction.messages.forward_invitation.ForwardInvitation
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing an invitation to be forwarded.

```
class Meta
```

Bases: *object*

Metadata for a forwarded invitation.

```
    handler_class = 'aries_cloudagent.protocols.introduction.handlers.forward_invitation'
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/introduction-service/0.1/forward_invitation'
```

```
    schema_class = 'ForwardInvitationSchema'
```

```
class aries_cloudagent.protocols.introduction.messages.forward_invitation.ForwardInvitationSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

ForwardInvitation request schema class.

```
class Meta
```

Bases: *object*

ForwardInvitation request schema metadata.

```
    model_class
```

alias of *ForwardInvitation*

```
    invitation = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None)>
```

```
    message = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
```

aries_cloudagent.protocols.introduction.messages.invitation module

Represents an invitation returned to the introduction service.

```

class aries_cloudagent.protocols.introduction.messages.invitation.Invitation(*,
                                                                           in-
                                                                           vi-
                                                                           ta-
                                                                           tion:
                                                                           aries_cloudagent.pr
                                                                           =
                                                                           None,
                                                                           mes-
                                                                           sage:
                                                                           str
                                                                           =
                                                                           None,
                                                                           **kwargs)

```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing an invitation returned to the introduction service.

```

class Meta
    Bases: object

    Metadata for an invitation.

    handler_class = 'aries_cloudagent.protocols.introduction.handlers.invitation_handle
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/introduction-service/0.1/invita
    schema_class = 'InvitationSchema'

```

```

class aries_cloudagent.protocols.introduction.messages.invitation.InvitationSchema(*args,
                                                                           **kwargs)

```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Invitation request schema class.

```

class Meta
    Bases: object

    Invitation request schema metadata.

    model_class
        alias of Invitation

    invitation = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=No
    message = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,

```

aries_cloudagent.protocols.introduction.messages.invitation_request module

Represents an request for an invitation from the introduction service.

```
class aries_cloudagent.protocols.introduction.messages.invitation_request.InvitationRequest
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing an invitation request.

```
class Meta
```

Bases: *object*

Metadata for an invitation request.

```
    handler_class = 'aries_cloudagent.protocols.introduction.handlers.invitation_request
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/introduction-service/0.1/invitation
```

```
    schema_class = 'InvitationRequestSchema'
```

```
class aries_cloudagent.protocols.introduction.messages.invitation_request.InvitationRequest
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Invitation request schema class.

```
class Meta
```

Bases: *object*

Invitation request schema metadata.

```
    model_class
```

alias of *InvitationRequest*

```
    message = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

```
    responder = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

Submodules

aries_cloudagent.protocols.introduction.base_service module

Introduction service base classes.

```
class aries_cloudagent.protocols.introduction.base_service.BaseIntroductionService(context:
                                                    aries_cloudagent.protocols.introduction.base_service.Context)
```

Bases: *abc.ABC*

Service handler for allowing connections to exchange invitations.

```
    return_invitation(target_connection_id: str, invitation: aries_cloudagent.protocols.introduction.messages.invitation.Invitation,
                    outbound_handler)
```

Handle the forwarding of an invitation to the responder.

Parameters

- **target_connection_id** – The ID of the connection sending the Invitation
- **invitation** – The received Invitation message
- **outbound_handler** – The outbound handler coroutine for sending a message

classmethod service_handler()

Quick accessor for conductor to use.

start_introduction(*init_connection_id: str, target_connection_id: str, outbound_handler, message: str = None*)

Start the introduction process between two connections.

Parameters

- **init_connection_id** – The connection initiating the request
- **target_connection_id** – The connection which is asked for an invitation
- **outbound_handler** – The outbound handler coroutine for sending a message
- **message** – The message to use when requesting the invitation

exception `aries_cloudagent.protocols.introduction.base_service.IntroductionError` (*args, error_code: str = None, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Generic introduction service error.

aries_cloudagent.protocols.introduction.demo_service module

Introduction service demo classes.

class `aries_cloudagent.protocols.introduction.demo_service.DemoIntroductionService`(*context: aries_cloudagent.protocols.introduction.messages.invitation.Invitation*)

Bases: `aries_cloudagent.protocols.introduction.base_service.BaseIntroductionService`

Service handler for allowing connections to exchange invitations.

RECORD_TYPE = 'introduction_record'

return_invitation(*target_connection_id: str, invitation: aries_cloudagent.protocols.introduction.messages.invitation.Invitation, outbound_handler*)

Handle the forwarding of an invitation to the responder.

Parameters

- **target_connection_id** – The ID of the connection sending the Invitation
- **invitation** – The received Invitation message
- **outbound_handler** – The outbound handler coroutine for sending a message

start_introduction(*init_connection_id: str, target_connection_id: str, message: str, outbound_handler*)

Start the introduction process between two connections.

Parameters

- **init_connection_id** – The connection initiating the request
- **target_connection_id** – The connection which is asked for an invitation
- **outbound_handler** – The outbound handler coroutine for sending a message
- **message** – The message to use when requesting the invitation

aries_cloudagent.protocols.introduction.message_types module

Message type identifiers for Introductions.

aries_cloudagent.protocols.introduction.routes module

Introduction service admin routes.

`aries_cloudagent.protocols.introduction.routes.introduction_start` (*request:*
<sphinx.ext.autodoc.importer._MockObject at 0x7f9e751605c0>)

Request handler for starting an introduction.

Parameters **request** – aiohttp request object

`aries_cloudagent.protocols.introduction.routes.register` (*app:*
<sphinx.ext.autodoc.importer._MockObject at 0x7f9e751605c0>)

Register routes.

3.1.7 aries_cloudagent.protocols.issue_credential package

Subpackages

aries_cloudagent.protocols.issue_credential.v1_0 package

Subpackages

aries_cloudagent.protocols.issue_credential.v1_0.handlers package

Submodules

aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_ack_handler module

Credential ack message handler.

class `aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_ack_handler.CredentialAckHandler`
 Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Message handler class for credential acks.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:*
`aries_cloudagent.messaging.responder.BaseResponder`)
 Message handler logic for credential acks.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler module

Credential issue message handler.

class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler.CredentialIssueHandler
 Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential offers.

handle (*context:* aries_cloudagent.messaging.request_context.RequestContext, *responder:* aries_cloudagent.messaging.responder.BaseResponder)
 Message handler logic for credential offers.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler module

Credential offer message handler.

class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler.CredentialOfferHandler
 Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential offers.

handle (*context:* aries_cloudagent.messaging.request_context.RequestContext, *responder:* aries_cloudagent.messaging.responder.BaseResponder)
 Message handler logic for credential offers.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler module

Credential proposal message handler.

class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler.CredentialProposalHandler
 Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential proposals.

handle (*context:* aries_cloudagent.messaging.request_context.RequestContext, *responder:* aries_cloudagent.messaging.responder.BaseResponder)
 Message handler logic for credential proposals.

Parameters

- **context** – proposal context

- **responder** – responder callback

aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler module

Credential request message handler.

class aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler
 Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for credential requests.

handle (context: *aries_cloudagent.messaging.request_context.RequestContext*, responder: *aries_cloudagent.messaging.responder.BaseResponder*)
 Message handler logic for credential requests.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.issue_credential.v1_0.messages package

Subpackages

aries_cloudagent.protocols.issue_credential.v1_0.messages.inner package

Submodules

aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview module

A credential preview inner object.

class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing a preview of an attribute.

class Meta

Bases: *object*

Attribute preview metadata.

schema_class = 'CredAttrSpecSchema'

b64_decoded_value () → str

Value, base64-decoded if applicable.

static list_plain (*plain: dict*)

Return a list of *CredAttrSpec* without MIME types from names/values.

Parameters **plain** – dict mapping names to values

Returns List of *CredAttrSpecs* with no MIME types

class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Attribute preview schema.

class **Meta**

Bases: *object*

Attribute preview schema metadata.

model_class

alias of *CredAttrSpec*

mime_type = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)

name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re

value = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r

class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing a credential preview inner object.

class **Meta**

Bases: *object*

Credential preview metadata.

message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/credential

schema_class = 'CredentialPreviewSchema'

attr_dict (*decode: bool = False*)

Return name:value pair per attribute.

Parameters **decode** – whether first to decode attributes with MIME type

mime_types ()

Return per-attribute mapping from name to MIME type.

Return empty dict if no attribute has MIME type.

class aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.C

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Credential preview schema.

```
class Meta
```

```
    Bases: object
```

```
    Credential preview schema metadata.
```

```
    model_class
```

```
        alias of CredentialPreview
```

```
    attributes = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=No
```

Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_ack` module

A credential ack message.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_ack.CredentialAck
```

```
    Bases: aries_cloudagent.messaging.ack.message.Ack
```

```
    Class representing a credential ack message.
```

```
    class Meta
```

```
        Bases: object
```

```
        Credential metadata.
```

```
        handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credenti
```

```
        message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/ack'
```

```
        schema_class = 'CredentialAckSchema'
```

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_ack.CredentialAckSchema
```

```
    Bases: aries_cloudagent.messaging.ack.message.AckSchema
```

```
    Credential ack schema.
```

```
    class Meta
```

```
        Bases: object
```

```
        Schema metadata.
```

```
        model_class
```

```
            alias of CredentialAck
```

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue` module

A credential content message.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.CredentialIssue:
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential.

```
class Meta
```

Bases: *object*

Credential metadata.

```
    handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler.CredentialIssueHandler'
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/issue-credential'
```

```
    schema_class = 'CredentialIssueSchema'
```

```
    indy_credential (index: int = 0)
```

Retrieve and decode indy credential from attachment.

Parameters *index* – ordinal in attachment list to decode and return (typically, list has length 1)

```
    classmethod wrap_indy_credential (indy_cred: dict) → AttachDecorator
        aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator
        Convert an indy credential offer to an attachment decorator.
```

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.CredentialIssueSchema:
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Credential schema.

```
class Meta
```

Bases: *object*

Credential schema metadata.

```
    model_class
```

alias of *CredentialIssue*

```
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_missing=None)>
```

```
    credentials_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_missing=None)>
```

aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer module

A credential offer content message.

class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.**CredentialOffer**

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential offer.

class Meta

Bases: *object*

CredentialOffer metadata.

handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler.CredentialOfferHandler'

message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/offer-credential'

schema_class = 'CredentialOfferSchema'

indy_offer (*index: int = 0*) → dict

Retrieve and decode indy offer from attachment.

Parameters **index** – ordinal in attachment list to decode and return (typically, list has length 1)

classmethod wrap_indy_offer (*indy_offer: dict*) → *aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator*

Convert an indy credential offer to an attachment decorator.

class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.**CredentialOfferSchema**

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Credential offer schema.

class Meta

Bases: *object*

Credential offer schema metadata.

```
model_class
    alias of CredentialOffer

comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
credential_preview = <fields.Nested(default=<marshmallow.missing>, attribute=None, val
offers_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate
```

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal` module

A credential proposal content message.

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.Credent
```



```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request.CredentialRequest
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a credential request.

```
class Meta
```

Bases: *object*

CredentialRequest metadata.

```
    handler_class = 'aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler.CredentialRequestHandler'
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/issue-credential/1.0/request-credential'
```

```
    schema_class = 'CredentialRequestSchema'
```

```
    indy_cred_req(index: int = 0)
```

Retrieve and decode indy credential request from attachment.

Parameters *index* – ordinal in attachment list to decode and return (typically, list has length 1)

```
    classmethod wrap_indy_cred_req(indy_cred_req: dict) → aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator
    Convert an indy credential request to an attachment decorator.
```

```
class aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request.CredentialRequestSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Credential request schema.

```
class Meta
```

Bases: *object*

Credential request schema metadata.

```
    model_class
```

alias of *CredentialRequest*

```
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)>
```

```
    requests_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)>
```

`aries_cloudagent.protocols.issue_credential.v1_0.models` package

Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange` module

Aries#0036 v1.0 credential exchange information with non-secrets storage.

class aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10Creden

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecord*

Represents an Aries#0036 credential exchange.

INITIATOR_EXTERNAL = 'external'

INITIATOR_SELF = 'self'

class Meta

Bases: *object*

CredentialExchange metadata.

schema_class = 'V10CredentialExchangeSchema'

RECORD_ID_NAME = 'credential_exchange_id'

RECORD_TYPE = 'credential_exchange_v10'

ROLE HOLDER = 'holder'

ROLE_ISSUER = 'issuer'

STATE_ACKED = 'credential_acked'

STATE_CREDENTIAL_RECEIVED = 'credential_received'

STATE_ISSUED = 'credential_issued'

STATE_OFFER_RECEIVED = 'offer_received'

STATE_OFFER_SENT = 'offer_sent'

STATE_PROPOSAL_RECEIVED = 'proposal_received'

STATE_PROPOSAL_SENT = 'proposal_sent'

STATE_REQUEST_RECEIVED = 'request_received'

STATE_REQUEST_SENT = 'request_sent'

TAG_NAMES = {'thread_id'}

WEBHOOK_TOPIC = 'issue_credential'

credential_exchange_id

Accessor for the ID associated with this exchange.

record_value

Accessor for the JSON record value generated for this credential exchange.

classmethod retrieve_by_connection_and_thread (*context*:

aries_cloudagent.config.injection_context.InjectionContext,

connection_id: *str,*

thread_id: *str*) *→*

aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange

Retrieve a credential exchange record by connection and thread ID.

class *aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange*

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecordSchema*

Schema to allow serialization/deserialization of credential exchange records.

class Meta

Bases: *object*

V10CredentialExchangeSchema metadata.

```
model_class
    alias of V10CredentialExchange

auto_issue = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None)
auto_offer = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=None)
connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_definition_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential_exchange_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
credential_offer = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_proposal_dict = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_request = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
credential_request_metadata = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
parent_thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
raw_credential = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
role = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
schema_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
```

Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.manager` module

Classes to manage credentials.

```
class aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager(context:
                                                                    aries_cloudagent.protocols.issue_credential.v1_0.manager.RequestContext)
```

Bases: `object`

Class for managing credentials.

context

Accessor for the current request context.

Returns The request context for this connection

```
create_offer (credential_exchange_record: aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchange,
              comment: str = None) → Tuple[aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchange,
              aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.CredentialOffer]
Create a credential offer, update credential exchange record.
```

Parameters

- **credential_exchange_record** – Credential exchange to create offer for

- **comment** – optional human-readable comment to set in offer message

Returns A tuple (credential exchange record, credential offer message)

```
create_proposal (connection_id: str, *, auto_offer: bool =
None, comment: str = None, credential_preview:
aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.CredentialPreview
= None, schema_id: str = None, schema_issuer_did: str =
None, schema_name: str = None, schema_version: str =
None, cred_def_id: str = None, issuer_did: str = None) →
aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange
Create a credential proposal.
```

Parameters

- **connection_id** – Connection to create proposal for
- **auto_offer** – Should this proposal request automatically be handled to offer a credential
- **comment** – Optional human-readable comment to include in proposal
- **credential_preview** – The credential preview to use to create the credential proposal
- **schema_id** – Schema id for credential proposal
- **schema_issuer_did** – Schema issuer DID for credential proposal
- **schema_name** – Schema name for credential proposal
- **schema_version** – Schema version for credential proposal
- **cred_def_id** – Credential definition id for credential proposal
- **issuer_did** – Issuer DID for credential proposal

Returns Resulting credential exchange record including credential proposal

```
create_request (credential_exchange_record: aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange,
holder_did: str) → Tuple[aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange,
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request.CredentialRequest]
Create a credential request.
```

Parameters

- **credential_exchange_record** – Credential exchange record for which to create request
- **holder_did** – holder DID

Returns A tuple (credential exchange record, credential request message)

```
issue_credential (credential_exchange_record: aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange,
*, comment: str = None, credential_values: dict) → Tuple[aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange,
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.CredentialIssue]
Issue a credential.
```

Parameters

- **credential_exchange_record** – The credential exchange record for which to issue a credential
- **comment** – optional human-readable comment pertaining to credential issue

- **credential_values** – dict of credential attribute {name: value} pairs

Returns (Updated credential exchange record, credential message)

Return type Tuple

prepare_send (*connection_id: str, credential_proposal: aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposal*) → Tuple[aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange, aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.CredentialOffer]
Set up a new credential exchange for an automated send.

Parameters

- **connection_id** – Connection to create offer for
- **credential_proposal** – The credential proposal with preview on attribute values to use if auto_issue is enabled

Returns A tuple of the new credential exchange record and credential offer message

receive_credential () → aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange
Receive a credential from an issuer.

Hold in storage potentially to be processed by controller before storing.

Returns Credential exchange record, retrieved and updated

receive_credential_ack () → aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange
Receive credential ack from holder.

Returns credential exchange record, retrieved and updated

receive_offer () → aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange
Receive a credential offer.

Returns The credential exchange record, updated

receive_proposal () → aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange
Receive a credential proposal from message in context on manager creation.

Returns The resulting credential exchange record, created

receive_request ()
Receive a credential request.

Parameters **credential_request_message** – Credential request to receive

Returns credential exchange record, retrieved and updated

store_credential (*credential_exchange_record: aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange, credential_id: str = None*) → Tuple[aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange, aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_ack.CredentialAck]
Store a credential in holder wallet; send ack to issuer.

Parameters **credential_exchange_record** – credential exchange record with credential to store and ack

Returns (Updated credential exchange record, credential ack message)

Return type Tuple

exception `aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManagerError` (`BaseError`)

Bases: `aries_cloudagent.core.error.BaseError`

Credential error.

`aries_cloudagent.protocols.issue_credential.v1_0.message_types` module

Message and inner object type identifiers for Connections.

`aries_cloudagent.protocols.issue_credential.v1_0.routes` module

Credential exchange admin routes.

class `aries_cloudagent.protocols.issue_credential.v1_0.routes.V10AttributeMimeTypeResults`

Bases: `marshmallow.schema.Schema`

Result schema for credential attribute MIME types by credential definition.

opts = `<marshmallow.schema.SchemaOpts object>`

class `aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialExchangeListRes`

Bases: `marshmallow.schema.Schema`

Result schema for Aries#0036 v1.0 credential exchange query.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialIssueRequestSch
```

Bases: `marshmallow.schema.Schema`

Request schema for sending credential issue admin message.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialOfferRequestSch
```

Bases: `marshmallow.schema.Schema`

Request schema for sending credential offer admin message.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProblemReportRec
```

Bases: `marshmallow.schema.Schema`

Request schema for sending problem report.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequest
```

Bases: `aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequestSchemaBase`

Request schema for sending credential proposal on mandatory proposal preview.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequest
```

Bases: `aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequestSchemaBase`

Request schema for sending credential proposal on optional proposal preview.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialProposalRequest
```

Bases: `marshmallow.schema.Schema`

Base class for request schema for sending credential proposal admin message.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.issue_credential.v1_0.routes.V10CredentialStoreRequestSch
```

Bases: `marshmallow.schema.Schema`

Request schema for sending a credential store admin message.

opts = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.attribute_mime_types_get (request:
<sphinx.ext.autodoc.Document object at 0x7f9e75ad9700>)
```

Request handler for getting credential attribute MIME types.

Parameters **request** – aiohttp request object

Returns The MIME types response

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_issue (request:
<sphinx.ext.autodoc.Document object at 0x7f9e75ad9700>)
```

Request handler for sending credential.

Parameters **request** – aiohttp request object

Returns The credential exchange record

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_list (request:
<sphinx.ext.autodoc.Document object at 0x7f9e75ad9700>)
```

Request handler for searching connection records.

Parameters **request** – aiohttp request object

Returns The connection list response

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_problem_report
```


Request handler for sending problem report.

Parameters `request` – aiohttp request object

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_remove(request:
<sphinx.ext.autodoc.directive.AutodocDirective object at
0x7f9e75ad97...
```

Request handler for removing a credential exchange record.

Parameters `request` – aiohttp request object

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_retrieve(request:
<sphinx.ext.autodoc.directive.AutodocDirective object at
0x7f9e75ad97...
```

Request handler for fetching single connection record.

Parameters `request` – aiohttp request object

Returns The credential exchange record

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send(request:
<sphinx.ext.autodoc.directive.AutodocDirective object at
0x7f9e75ad97...
```

Request handler for sending credential from issuer to holder from attr values.

If both issuer and holder are configured for automatic responses, the operation ultimately results in credential issue; otherwise, the result waits on the first response not automated; the credential exchange record retains state regardless.

Parameters `request` – aiohttp request object

Returns The credential exchange record

```
aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_bound_off...
```

Request handler for sending bound credential offer.

A holder initiates this sequence with a credential proposal; this message responds with an offer bound to the proposal.

Parameters `request` – aiohttp request object

Returns The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_free_offer`

Request handler for sending free credential offer.

An issuer initiates a such a credential offer, free from any holder-initiated corresponding credential proposal with preview.

Parameters `request` – aiohttp request object

Returns The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_proposal` (*request: aiohttp*

Request handler for sending credential proposal.

Parameters `request` – aiohttp request object

Returns The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_send_request` (*request: aiohttp*

Request handler for sending credential request.

Parameters `request` – aiohttp request object

Returns The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.credential_exchange_store` (*request: aiohttp*

Request handler for storing credential.

Parameters `request` – aiohttp request object

Returns The credential exchange record

`aries_cloudagent.protocols.issue_credential.v1_0.routes.register` (*app: sphinx.ext.autodoc.importer._MockC*

Register routes.

Submodules

aries_cloudagent.protocols.issue_credential.message_types module

All issue_credential message types.

aries_cloudagent.protocols.issue_credential.routes module

All issue_credential routes.

```
aries_cloudagent.protocols.issue_credential.routes.register(app:
                                                         <sphinx.ext.autodoc.importer._MockObject
                                                         object                at
                                                         0x7f9e75ad9780>)
```

Register routes.

3.1.8 aries_cloudagent.protocols.present_proof package

Subpackages

aries_cloudagent.protocols.present_proof.v1_0 package

Subpackages

aries_cloudagent.protocols.present_proof.v1_0.handlers package

Submodules

aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler module

Presentation ack message handler.

```
class aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler.PresentationAckHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for presentation acks.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
```

Message handler logic for presentation acks.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_handler module

Presentation message handler.

```
class aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_handler.PresentationHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for presentations.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
Message handler logic for presentations.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_proposal_handler module

Presentation proposal message handler.

class *aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_proposal_handler*
Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for presentation proposals.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
Message handler logic for presentation proposals.

Parameters

- **context** – proposal context
- **responder** – responder callback

aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_request_handler module

Presentation request message handler.

class *aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_request_handler*
Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Message handler class for Aries#0037 v1.0 presentation requests.

handle (*context:* *aries_cloudagent.messaging.request_context.RequestContext*, *responder:* *aries_cloudagent.messaging.responder.BaseResponder*)
Message handler logic for Aries#0037 v1.0 presentation requests.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.present_proof.v1_0.messages package

Subpackages

aries_cloudagent.protocols.present_proof.v1_0.messages.inner package

Submodules

aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview module

A presentation preview inner object.

class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.**Pre**

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing an attribute specification within a presentation preview.

class Meta

Bases: *object*

Attr spec metadata.

schema_class = 'PresAttrSpecSchema'

class Posture

Bases: *enum.Enum*

Attribute posture: self-attested, revealed claim or unrevealed claim.

REVEALED_CLAIM = 1

SELF_ATTESTED = 0

UNREVEALED_CLAIM = 2

b64_decoded_value() → str

Value, base64-decoded if applicable.

static list_plain(plain: dict, cred_def_id: str, referent: str = None)

Return a list of *PresAttrSpec* on input cred def id.

Parameters

- **plain** – dict mapping names to values
- **cred_def_id** – credential definition identifier to specify
- **referent** – single referent to use, omit for none

Returns List of PresAttrSpec on input cred def id with no MIME types

posture

self-attested, revealed claim, or unrevealed claim.

Type Attribute posture

satisfies (*pred_spec: aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.PresPredSpec*)

Whether current specified attribute satisfied input specified predicate.

class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.**PresAttrSpec**

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Attribute specification schema.

class Meta

Bases: *object*

Attribute specification schema metadata.

model_class

alias of *PresAttrSpec*

cred_def_id = *<fields.String(default=<marshmallow.missing>, attribute=None, validate=<*

mime_type = *<fields.String(default=<marshmallow.missing>, attribute=None, validate=None*

name = *<fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re*

referent = *<fields.String(default=<marshmallow.missing>, attribute=None, validate=None*

value = *<fields.String(default=<marshmallow.missing>, attribute=None, validate=None, r*

class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.**PresPredSpec**

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing a predicate specification within a presentation preview.

class Meta

Bases: *object*

Pred spec metadata.

schema_class = *'PresPredSpecSchema'*

class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.**PresPredSpecSchema**

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Predicate specification schema.

```

class Meta
    Bases: object

    Predicate specification schema metadata.

    model_class
        alias of PresPredSpec

    cred_def_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<
    name = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, re
    predicate = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<ar
    threshold = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=No
class aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.Pres

```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing presentation preview.

```

class Meta
    Bases: object

    Presentation preview metadata.

    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/presentation-
    schema_class = 'PresentationPreviewSchema'

    has_attr_spec(cred_def_id: str, name: str, value: str) → bool
        Return whether preview contains given attribute specification.

```

Parameters

- **cred_def_id** – credential definition identifier
- **name** – attribute name
- **value** – attribute value

Returns Whether preview contains matching attribute specification.

indy_proof_request (*name: str = None, version: str = None, nonce: str = None, ledger: aries_cloudagent.ledger.indy.IndyLedger = None, timestamps: Mapping[str, int] = None*) → dict

Return indy proof request corresponding to presentation preview.

Typically the verifier turns the proof preview into a proof request.

Parameters

- **name** – for proof request
- **version** – version for proof request
- **nonce** – nonce for proof request
- **ledger** – ledger with credential definitions, to check for revocation support
- **timestamps** – dict mapping cred def ids to non-revocation timestamps to use (default current time where applicable)

Returns Indy proof request dict.

class `aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.PresentationPreview`

Bases: `aries_cloudagent.messaging.models.base.BaseModelSchema`

Presentation preview schema.

class **Meta**

Bases: `object`

Presentation preview schema metadata.

model_class

alias of `PresentationPreview`

attributes = `<fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None)`

predicates = `<fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None)`

Submodules

`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation` module

A (proof) presentation content message.


```
class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation.Presentation(_id: str,
                                           *,
                                           comment: str,
                                           = None,
                                           pre-
                                           sen-
                                           ta-
                                           tions: Se-
                                           quen
                                           = None,
                                           **kw)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a (proof) presentation.

```
class Meta
```

Bases: *object*

Presentation metadata.

```
handler_class = 'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation'
```

```
message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/presentation'
```

```
schema_class = 'PresentationSchema'
```

```
indy_proof(index: int = 0)
```

Retrieve and decode indy proof from attachment.

Parameters **index** – ordinal in attachment list to decode and return (typically, list has length 1)

```
class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation.PresentationSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

(Proof) presentation schema.

```
class Meta
```

Bases: *object*

Presentation schema metadata.

```
model_class
```

alias of *Presentation*

```
comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

```
presentations_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, v
```

aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack module

Represents an explicit RFC 15 ack message, adopted into present-proof protocol.

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack.**PresentationAck**

Bases: *aries_cloudagent.messaging.ack.message.Ack*

Base class representing an explicit ack message for present-proof protocol.

class Meta

Bases: *object*

PresentationAck metadata.

handler_class = 'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler.PresentationAckHandler'

message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/ack'

schema_class = 'PresentationAckSchema'

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack.**PresentationAckSchema**

Bases: *aries_cloudagent.messaging.ack.message.AckSchema*

Schema for PresentationAck class.

class Meta

Bases: *object*

PresentationAck schema metadata.

model_class

alias of *PresentationAck*

aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal module

A presentation proposal content message.

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal.**PresentationProposal**

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a presentation proposal.

```

class Meta
    Bases: object
    PresentationProposal metadata.
    handler_class = 'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_proposal'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/propose-presentation'
    schema_class = 'PresentationProposalSchema'

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal.PresentationProposal
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Presentation proposal schema.

class Meta
    Bases: object
    Presentation proposal schema metadata.
    model_class
        alias of PresentationProposal
    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_not_none=' ')>
    presentation_proposal = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_not_none=' ')>

```

aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request module

A presentation request content message.

```

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequest

```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a presentation request.

```

class Meta
    Bases: object
    PresentationRequest metadata.
    handler_class = 'aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_request'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/present-proof/1.0/request-presentation'

```

```
    schema_class = 'PresentationRequestSchema'

    indy_proof_request (index: int = 0)
        Retrieve and decode indy proof request from attachment.

        Parameters index – ordinal in attachment list to decode and return (typically, list has length
            1)

class aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequestSchema:
    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    Presentation request schema.

    class Meta:
        Bases: object
        Presentation request schema metadata.

    model_class
        alias of PresentationRequest

    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_not_none=None)>
    request_presentations_attach = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None, if_not_none=None)>
```

aries_cloudagent.protocols.present_proof.v1_0.models package

Submodules

aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange module

Aries#0037 v1.0 presentation exchange information with non-secrets storage.

```
class aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10Present
```

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecord*

Represents an Aries#0037 v1.0 presentation exchange.

```
INITIATOR_EXTERNAL = 'external'
INITIATOR_SELF = 'self'

class Meta
    Bases: object
    V10PresentationExchange metadata.
    schema_class = 'V10PresentationExchangeSchema'
RECORD_ID_NAME = 'presentation_exchange_id'
RECORD_TYPE = 'presentation_exchange_v10'
ROLE_PROVER = 'prover'
ROLE_VERIFIER = 'verifier'
STATE_PRESENTATION_ACKED = 'presentation_acked'
STATE_PRESENTATION_RECEIVED = 'presentation_received'
STATE_PRESENTATION_SENT = 'presentation_sent'
STATE_PROPOSAL_RECEIVED = 'proposal_received'
STATE_PROPOSAL_SENT = 'proposal_sent'
STATE_REQUEST_RECEIVED = 'request_received'
STATE_REQUEST_SENT = 'request_sent'
STATE_VERIFIED = 'verified'
TAG_NAMES = {'thread_id'}
WEBHOOK_TOPIC = 'present_proof'

presentation_exchange_id
    Accessor for the ID associated with this exchange.

record_value
    Accessor for JSON record value generated for this presentation exchange.
```

```
class aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10Present
```

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecordSchema*

Schema for de/serialization of v1.0 presentation exchange records.

```
class Meta
    Bases: object
    V10PresentationExchangeSchema metadata.
    model_class
        alias of V10PresentationExchange

auto_present = <fields.Boolean(default=<marshmallow.missing>, attribute=None, validate=
connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=Non
initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<On
```

```

presentation = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
presentation_exchange_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
presentation_proposal_dict = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
presentation_request = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)
role = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf(choices=('holder', 'issuer', 'verifier'), error='Invalid role'))>
state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)
thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, if_present=True)
verified = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf(choices=('true', 'false'), error='Invalid verified status'))>

```

aries_cloudagent.protocols.present_proof.v1_0.util package

Submodules

aries_cloudagent.protocols.present_proof.v1_0.util.indy module

Utilities for dealing with indy conventions.

```
aries_cloudagent.protocols.present_proof.v1_0.util.indy.indy_proof_req_preview2indy_request
```

Build indy requested-credentials structure.

Given input proof request and presentation preview, use credentials in holder's wallet to build indy requested credentials structure for input to proof creation.

Parameters

- **indy_proof_request** – indy proof request
- **pres_preview** – preview from presentation proposal, if applicable
- **holder** – holder injected into current context

Submodules

aries_cloudagent.protocols.present_proof.v1_0.manager module

Classes to manage presentations.

```
class aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager(context: Context, wallet: Wallet,
                                                                              aries_cloudagent.protocols.present_proof.v1_0.manager.indy_util: indy_util)
```

Bases: `object`

Class for managing presentations.

context

Accessor for the current request context.

Returns The injection context for this presentation manager

create_bound_request (*presentation_exchange_record: aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange_record, name: str = None, version: str = None, nonce: str = None, comment: str = None*)

Create a presentation request bound to a proposal.

Parameters

- **presentation_exchange_record** – Presentation exchange record for which to create presentation request
- **name** – name to use in presentation request (None for default)
- **version** – version to use in presentation request (None for default)
- **nonce** – nonce to use in presentation request (None to generate)
- **comment** – Optional human-readable comment pertaining to request creation

Returns A tuple (updated presentation exchange record, presentation request message)

create_exchange_for_proposal (*connection_id: str, presentation_proposal_message: aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal.PresentationProposal, auto_present: bool = None*)

Create a presentation exchange record for input presentation proposal.

Parameters

- **connection_id** – connection identifier
- **presentation_proposal_message** – presentation proposal to serialize to exchange record
- **auto_present** – whether to present proof upon receiving proof request (default to configuration setting)

Returns Presentation exchange record, created

create_exchange_for_request (*connection_id: str, presentation_request_message: aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequest*)

Create a presentation exchange record for input presentation request.

Parameters

- **connection_id** – connection identifier
- **presentation_request_message** – presentation request to use in creating exchange record, extracting indy proof request and thread id

Returns Presentation exchange record, updated

create_presentation (*presentation_exchange_record: aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange_record, requested_credentials: dict, comment: str = None*)

Create a presentation.

Parameters

- **presentation_exchange_record** – Record to update
- **requested_credentials** – Indy formatted requested_credentials
- **comment** – optional human-readable comment

Example *requested_credentials* format:


```
{
  "self_attested_attributes": {
    "j233ffbc-bd35-49b1-934f-51e083106f6d": "value"
  },
  "requested_attributes": {
    "6253ffbb-bd35-49b3-934f-46e083106f6c": {
      "cred_id": "5bfa40b7-062b-4ae0-a251-a86c87922c0e",
      "revealed": true
    }
  },
  "requested_predicates": {
    "bfc8a97d-60d3-4f21-b998-85eeabe5c8c0": {
      "cred_id": "5bfa40b7-062b-4ae0-a251-a86c87922c0e"
    }
  }
}
```

Returns A tuple (updated presentation exchange record, presentation message)

receive_presentation()

Receive a presentation, from message in context on manager creation.

Returns presentation exchange record, retrieved and updated

receive_presentation_ack()

Receive a presentation ack, from message in context on manager creation.

Returns presentation exchange record, retrieved and updated

receive_proposal()

Receive a presentation proposal from message in context on manager creation.

Returns Presentation exchange record, created

receive_request (*presentation_exchange_record: aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exch*)

Receive a presentation request.

Parameters **presentation_exchange_record** – presentation exchange record with request to receive

Returns The presentation_exchange_record, updated

send_presentation_ack (*presentation_exchange_record: aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exch*)

Send acknowledgement of presentation receipt.

Parameters **presentation_exchange_record** – presentation exchange record with thread id

verify_presentation (*presentation_exchange_record: aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exch*)

Verify a presentation.

Parameters **presentation_exchange_record** – presentation exchange record with presentation request and presentation to verify

Returns presentation record, updated

exception `aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManagerError` (*a
er
ro
st
=
Ne
**

Bases: `aries_cloudagent.core.error.BaseError`

Presentation error.

`aries_cloudagent.protocols.present_proof.v1_0.message_types` module

Message and inner object type identifiers for Connections.

`aries_cloudagent.protocols.present_proof.v1_0.routes` module

Admin routes for presentations.

class `aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqAttrSpecSchema` (*,
only=
ex-
clude=
many=
con-
text=N
load_c
dump_
par-
tial=F
un-
known

Bases: `marshmallow.schema.Schema`

Schema for attribute specification in indy proof request.

opts = `<marshmallow.schema.SchemaOpts object>`

class `aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqNonRevoked` (*,
only=None,
ex-
clude=(),
many=False
con-
text=None,
load_only=(
dump_only=
par-
tial=False,
un-
known=Non

Bases: `marshmallow.schema.Schema`

Non-revocation times specification in indy proof request.

opts = `<marshmallow.schema.SchemaOpts object>`

```

class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqPredSpecSchema(*,
    only=None,
    exclude=None,
    many=False,
    text=None,
    load_only=None,
    dump_only=None,
    partial=False,
    unknown=None)

    Bases: marshmallow.schema.Schema
    Schema for predicate specification in indy proof request.
    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofReqSpecRestrictionsSchema(*,
    only=None,
    exclude=None,
    many=False,
    text=None,
    load_only=None,
    dump_only=None,
    partial=False,
    unknown=None)

    Bases: marshmallow.schema.Schema
    Schema for restrictions in attr or pred specifier indy proof request.
    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyProofRequestSchema(*,
    only=None,
    exclude=None,
    many=False,
    text=None,
    load_only=None,
    dump_only=None,
    partial=False,
    unknown=None)

    Bases: marshmallow.schema.Schema
    Schema for indy proof request.
    opts = <marshmallow.schema.SchemaOpts object>

```

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyRequestedCredsRequestedAttr:
```

Bases: `marshmallow.schema.Schema`

Schema for requested attributes within indy requested credentials structure.

```
    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.IndyRequestedCredsRequestedPred:
```

Bases: `marshmallow.schema.Schema`

Schema for requested predicates within indy requested credentials structure.

```
    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationExchangeListScher
```

Bases: `marshmallow.schema.Schema`

Result schema for an Aries#0037 v1.0 presentation exchange query.

```
    opts = <marshmallow.schema.SchemaOpts object>
```

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationProposalRequestS
```

Bases: `marshmallow.schema.Schema`

Request schema for sending a presentation proposal admin message.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationRequestRequestSch
```

Bases: `marshmallow.schema.Schema`

Request schema for sending a proof request.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.present_proof.v1_0.routes.V10PresentationRequestSchema (*,
```

only
ex-
clud
man
com
text
load
dum
par
tial
un-
know

Bases: `marshmallow.schema.Schema`

Request schema for sending a presentation.

opts = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_create_request (request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for creating a free presentation request.

The presentation request will not be bound to any proposal or existing connection.

Parameters `request` – aiohttp request object

Returns The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_credentials_list (request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for searching applicable credential records.

Parameters `request` – aiohttp request object

Returns The credential list response

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_list (request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for searching presentation exchange records.

Parameters `request` – aiohttp request object

Returns The presentation exchange list response

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_remove (request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for removing a presentation exchange record.

Parameters `request` – aiohttp request object

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_retrieve (request: aiohttp.Request) -> Dict[str, Any]
```

Request handler for fetching a single presentation exchange record.

Parameters `request` – aiohttp request object

Returns The presentation exchange record response

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_bound_request
```

Request handler for sending a presentation request free from any proposal.

Parameters `request` – aiohttp request object

Returns The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_free_request
```

Request handler for sending a presentation request free from any proposal.

Parameters `request` – aiohttp request object

Returns The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_presentation
```

Request handler for sending a presentation.

Parameters `request` – aiohttp request object

Returns The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_send_proposal (request)
```

Request handler for sending a presentation proposal.

Parameters `request` – aiohttp request object

Returns The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.presentation_exchange_verify_presentation
```

Request handler for verifying a presentation request.

Parameters `request` – aiohttp request object

Returns The presentation exchange details

```
aries_cloudagent.protocols.present_proof.v1_0.routes.register(app:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object          at
                                                                0x7f9e766c30b8>)
```

Register routes.

Submodules

aries_cloudagent.protocols.present_proof.message_types module

All present_proof message types.

aries_cloudagent.protocols.present_proof.routes module

All present_proof routes.

```
aries_cloudagent.protocols.present_proof.routes.register(app:
                                                                <sphinx.ext.autodoc.importer._MockObject
                                                                object          at
                                                                0x7f9e766c30b8>)
```

Register routes.

3.1.9 aries_cloudagent.protocols.presentations package

Subpackages

aries_cloudagent.protocols.presentations.handlers package

Submodules

aries_cloudagent.protocols.presentations.handlers.credential_presentation_handler module

Basic message handler.

```
class aries_cloudagent.protocols.presentations.handlers.credential_presentation_handler.Cred
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler
```

Message handler class for credential presentations.

```
handle (context:          aries_cloudagent.messaging.request_context.RequestContext,      responder:
        aries_cloudagent.messaging.responder.BaseResponder)
```

Message handler logic for credential presentations.

Parameters

- **context** – request context
- **responder** – responder callback

aries_cloudagent.protocols.presentations.handlers.presentation_request_handler module

Basic message handler.

class `aries_cloudagent.protocols.presentations.handlers.presentation_request_handler.PresentationRequestHandler`
 Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Message handler class for presentation requests.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
 Message handler logic for presentation requests.

Parameters

- **context** – request context
- **responder** – responder callback

`aries_cloudagent.protocols.presentations.messages` package

Submodules

`aries_cloudagent.protocols.presentations.messages.credential_presentation` module

A credential presentation message.

class `aries_cloudagent.protocols.presentations.messages.credential_presentation.CredentialPresentation`

Bases: `aries_cloudagent.messaging.agent_message.AgentMessage`

Class representing a credential presentation.

class `Meta`

Bases: `object`

CredentialPresentation metadata.

handler_class = `'aries_cloudagent.protocols.presentations.handlers.credential_presentation_handler.CredentialPresentationHandler'`

message_type = `'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/credential-presentation/0.1/credential-presentation'`

schema_class = `'CredentialPresentationSchema'`

class `aries_cloudagent.protocols.presentations.messages.credential_presentation.CredentialPresentationSchema`

Bases: `aries_cloudagent.messaging.agent_message.AgentMessageSchema`

CredentialPresentation schema.

class `Meta`

Bases: `object`

CredentialPresentationSchema metadata.

model_class

alias of `CredentialPresentation`

```
comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
presentation = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

`aries_cloudagent.protocols.presentations.messages.presentation_request` module

A presentation request content message.

```
class aries_cloudagent.protocols.presentations.messages.presentation_request.PresentationRequest
```

Bases: `aries_cloudagent.messaging.agent_message.AgentMessage`

Class representing a presentation request.

```
class Meta
```

Bases: `object`

PresentationRequest metadata.

```
handler_class = 'aries_cloudagent.protocols.presentations.handlers.presentation_request
```

```
message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/credential-presentation/0.1/presentation
```

```
schema_class = 'PresentationRequestSchema'
```

```
class aries_cloudagent.protocols.presentations.messages.presentation_request.PresentationRequestSchema
```

Bases: `aries_cloudagent.messaging.agent_message.AgentMessageSchema`

PresentationRequest schema.

```
class Meta
```

Bases: `object`

PresentationRequestSchema metadata.

```
model_class
```

alias of `PresentationRequest`

```
comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

```
request = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

`aries_cloudagent.protocols.presentations.models` package

Submodules

`aries_cloudagent.protocols.presentations.models.presentation_exchange` module

Handle presentation exchange information interface with non-secrets storage.

```
class aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationEx
```

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecord*

Represents a presentation exchange.

INITIATOR_EXTERNAL = 'external'

INITIATOR_SELF = 'self'

LOG_STATE_FLAG = 'debug.presentations'

class Meta

Bases: *object*

PresentationExchange metadata.

schema_class = 'PresentationExchangeSchema'

RECORD_ID_NAME = 'presentation_exchange_id'

RECORD_TYPE = 'presentation_exchange'

STATE_PRESENTATION_RECEIVED = 'presentation_received'

STATE_PRESENTATION_SENT = 'presentation_sent'

STATE_REQUEST_RECEIVED = 'request_received'

STATE_REQUEST_SENT = 'request_sent'

STATE_VERIFIED = 'verified'

TAG_NAMES = {'thread_id'}

WEBHOOK_TOPIC = 'presentations'

presentation_exchange_id

Accessor for the ID associated with this exchange.

record_value

Accessor for JSON record value generated for this presentation exchange.

class *aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange*

Bases: *aries_cloudagent.messaging.models.base_record.BaseRecordSchema*

Schema for serialization/deserialization of presentation exchange records.

class Meta

Bases: *object*

PresentationExchangeSchema metadata.

model_class

alias of *PresentationExchange*

connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>

error_msg = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>

initiator = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>

presentation = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)>

presentation_exchange_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>

presentation_request = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=None)>

state = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, required=True)>

thread_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>

```
verified = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)
```

Submodules

aries_cloudagent.protocols.presentations.manager module

Classes to manage presentations.

class aries_cloudagent.protocols.presentations.manager.**PresentationManager** (*context:* aries_cloudagent.conf...

Bases: `object`

Class for managing presentations.

context

Accessor for the current request context.

Returns The injection context for this presentation manager

create_presentation (*presentation_exchange_record: aries_cloudagent.protocols.presentations.models.presentation_exch...*
requested_credentials: dict)

Receive a presentation request.

Parameters

- **presentation_exchange_record** – Record to update
- **requested_credentials** – Indy formatted requested_credentials

i.e.,

```
{
  "self_attested_attributes": {
    "j233ffbc-bd35-49b1-934f-51e083106f6d": "value"
  },
  "requested_attributes": {
    "6253ffbb-bd35-49b3-934f-46e083106f6c": {
      "cred_id": "5bfa40b7-062b-4ae0-a251-a86c87922c0e",
      "revealed": true
    }
  },
  "requested_predicates": {
    "bfc8a97d-60d3-4f21-b998-85eeabe5c8c0": {
      "cred_id": "5bfa40b7-062b-4ae0-a251-a86c87922c0e"
    }
  }
}
```

create_request (*name: str, version: str, requested_attributes: list, requested_predicates: list, con-*
nection_id: str)

Create a proof request.

receive_presentation (*presentation: dict, thread_id: str*)

Receive a presentation request.

receive_request (*presentation_request_message: aries_cloudagent.protocols.presentations.messages.presentation_request...*
connection_id: str)

Receive a presentation request.

Parameters **presentation_request_message** – Presentation message to receive

verify_presentation (*presentation_exchange_record: aries_cloudagent.protocols.presentations.models.presentation_exchange_record*)
 Verify a presentation request.

exception `aries_cloudagent.protocols.presentations.manager.PresentationManagerError` (*args, error_code: str, message: str, status_code: int, **kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Presentation error.

aries_cloudagent.protocols.presentations.message_types module

Message type identifiers for presentations.

aries_cloudagent.protocols.presentations.routes module

Admin routes for presentations.

class `aries_cloudagent.protocols.presentations.routes.PresentationExchangeListSchema` (*, only=None, exclude=None, many=False, context=None, load_only=None, dump_only=None, partial=False, if_equal=None, unknown=None)

Bases: `marshmallow.schema.Schema`

Result schema for a presentation exchange query.

opts = `<marshmallow.schema.SchemaOpts object>`

class `aries_cloudagent.protocols.presentations.routes.PresentationRequestRequestSchema` (*, only=None, exclude=None, many=False, context=None, load_only=None, dump_only=None, partial=False, if_equal=None, unknown=None)

Bases: `marshmallow.schema.Schema`

Request schema for sending a proof request.

```

class RequestedAttribute(*, only=None, exclude=(), many=False, context=None,
                        load_only=(), dump_only=(), partial=False, unknown=None)
    Bases: marshmallow.schema.Schema
    RequestedAttribute model.

    opts = <marshmallow.schema.SchemaOpts object>

class RequestedPredicate(*, only=None, exclude=(), many=False, context=None,
                        load_only=(), dump_only=(), partial=False, unknown=None)
    Bases: marshmallow.schema.Schema
    RequestedPredicate model.

    opts = <marshmallow.schema.SchemaOpts object>

    opts = <marshmallow.schema.SchemaOpts object>

class aries_cloudagent.protocols.presentations.routes.SendPresentationRequestSchema(*,
only=None, exclude=(), many=False, context=None, load_only=(), dump_only=(), partial=False, unknown=None)
    Bases: marshmallow.schema.Schema
    Request schema for sending a presentation.

    opts = <marshmallow.schema.SchemaOpts object>

aries_cloudagent.protocols.presentations.routes.presentation_exchange_create_request(request: aiohttp.Request)
    Request handler for creating a presentation request.

    Parameters request – aiohttp request object
    Returns The presentation exchange details.

aries_cloudagent.protocols.presentations.routes.presentation_exchange_credentials_list(request: aiohttp.Request)
    Request handler for searching applicable credential records.

    Parameters request – aiohttp request object
    Returns The credential list response

```

`aries_cloudagent.protocols.presentations.routes.presentation_exchange_list` (*request:*
<sphinx.ext.autodoc.in
ob-
ject
at
0x7f9e75449ba8>)

Request handler for searching presentation exchange records.

Parameters `request` – aiohttp request object

Returns The presentation exchange list response

`aries_cloudagent.protocols.presentations.routes.presentation_exchange_remove` (*request:*
<sphinx.ext.autodoc
ob-
ject
at
0x7f9e75449ba8>)

Request handler for removing a presentation exchange record.

Parameters `request` – aiohttp request object

`aries_cloudagent.protocols.presentations.routes.presentation_exchange_retrieve` (*request:*
<sphinx.ext.autodoc
ob-
ject
at
0x7f9e75449ba8>)

Request handler for fetching a single presentation exchange record.

Parameters `request` – aiohttp request object

Returns The presentation exchange record response

`aries_cloudagent.protocols.presentations.routes.presentation_exchange_send_credential_pres`

Request handler for sending a credential presentation.

Parameters `request` – aiohttp request object

Returns The presentation exchange details.

`aries_cloudagent.protocols.presentations.routes.presentation_exchange_send_request` (*request:*
<sphinx.ex
ob-
ject
at
0x7f9e7544

Request handler for creating and sending a presentation request.

Parameters `request` – aiohttp request object

Returns The presentation exchange details.

`aries_cloudagent.protocols.presentations.routes.presentation_exchange_verify_credential_pr`

Request handler for verifying a presentation request.

Parameters `request` – aiohttp request object

Returns The presentation exchange details.

`aries_cloudagent.protocols.presentations.routes.register` (*app:*
<sphinx.ext.autodoc.importer._MockObject
object *at*
0x7f9e75449ba8>)

Register routes.

3.1.10 `aries_cloudagent.protocols.problem_report` package

Submodules

`aries_cloudagent.protocols.problem_report.handler` module

Generic problem report handler.

class `aries_cloudagent.protocols.problem_report.handler.ProblemReportHandler`
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Problem report handler class.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:*
`aries_cloudagent.messaging.responder.BaseResponder`)
Handle problem report message.

Parameters

- **context** – Request context
- **responder** – Responder used to reply

`aries_cloudagent.protocols.problem_report.message` module

Represents a generic problem report message.

```

class aries_cloudagent.protocols.problem_report.message.ProblemReport(*,
                                                                    msg_catalog:
                                                                    str =
                                                                    None,
                                                                    lo-
                                                                    cale:
                                                                    str =
                                                                    None,
                                                                    ex-
                                                                    plain_ltxt:
                                                                    str =
                                                                    None,
                                                                    ex-
                                                                    plain_l10n:
                                                                    Map-
                                                                    ping[str,
                                                                    str] =
                                                                    None,
                                                                    prob-
                                                                    lem_items:
                                                                    Se-
                                                                    quence[Mapping[str,
                                                                    str]]
                                                                    =
                                                                    None,
                                                                    who_retries:
                                                                    str =
                                                                    None,
                                                                    fix_hint_ltxt:
                                                                    Map-
                                                                    ping[str,
                                                                    str] =
                                                                    None,
                                                                    im-
                                                                    pact:
                                                                    str =
                                                                    None,
                                                                    where:
                                                                    str =
                                                                    None,
                                                                    time_noticed:
                                                                    str =
                                                                    None,
                                                                    track-
                                                                    ing_uri:
                                                                    str =
                                                                    None,
                                                                    es-
                                                                    cala-
                                                                    tion_uri:
                                                                    str =
                                                                    None,
                                                                    **kwargs)

```

Bases: `aries_cloudagent.messaging.agent_message.AgentMessage`

Base class representing a generic problem report message.

```
class Meta
    Bases: object

    Problem report metadata.

    handler_class = 'aries_cloudagent.protocols.problem_report.handler.ProblemReportHan
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/notification/1.0/problem-report
    schema_class = 'ProblemReportSchema'
```

```
class aries_cloudagent.protocols.problem_report.message.ProblemReportSchema(*args,
                                                                              **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Schema for ProblemReport base class.

```
class Meta
    Bases: object

    Problem report schema metadata.

    model_class
        alias of ProblemReport

    escalation_uri = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
    explain_l10n = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=N
    explain_ltxt = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
    fix_hint_ltxt = <fields.Dict(default=<marshmallow.missing>, attribute=None, validate=N
    impact = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<OneOf
    locale = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    msg_catalog = <fields.String(default=<marshmallow.missing>, attribute=None, validate=N
    problem_items = <fields.List(default=<marshmallow.missing>, attribute=None, validate=N
    time_noticed = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
    tracking_uri = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
    where = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<Regexp
    who_retries = <fields.String(default=<marshmallow.missing>, attribute=None, validate=<
```

aries_cloudagent.protocols.problem_report.message_types module

Message type identifiers for problem reports.

3.1.11 aries_cloudagent.protocols.routing package

Subpackages

aries_cloudagent.protocols.routing.handlers package

Submodules

`aries_cloudagent.protocols.routing.handlers.forward_handler` module

Handler for incoming forward messages.

class `aries_cloudagent.protocols.routing.handlers.forward_handler.ForwardHandler`
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming forward messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.routing.handlers.route_query_request_handler` module

Handler for incoming route-query-request messages.

class `aries_cloudagent.protocols.routing.handlers.route_query_request_handler.RouteQueryRequestHandler`
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming route-query-request messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.routing.handlers.route_query_response_handler` module

Handler for incoming route-query-response messages.

class `aries_cloudagent.protocols.routing.handlers.route_query_response_handler.RouteQueryResponseHandler`
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming route-query-response messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.routing.handlers.route_update_request_handler` module

Handler for incoming route-update-request messages.

class `aries_cloudagent.protocols.routing.handlers.route_update_request_handler.RouteUpdateRequestHandler`
Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Handler for incoming route-update-request messages.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
Message handler implementation.

`aries_cloudagent.protocols.routing.handlers.route_update_response_handler` module

Handler for incoming route-update-response messages.

```
class aries_cloudagent.protocols.routing.handlers.route_update_response_handler.RouteUpdateResponseHandler
    Bases: aries_cloudagent.messaging.base_handler.BaseHandler

    Handler for incoming route-update-response messages.

    handle (context: aries_cloudagent.messaging.request_context.RequestContext, responder:
             aries_cloudagent.messaging.responder.BaseResponder)
        Message handler implementation.
```

aries_cloudagent.protocols.routing.messages package

Submodules

aries_cloudagent.protocols.routing.messages.forward module

Represents a forward message.

```
class aries_cloudagent.protocols.routing.messages.forward.Forward(*, to: str =
                                                                    None, msg:
                                                                    Union[dict,
                                                                    str] = None,
                                                                    **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Represents a request to forward a message to a connected agent.

```
class Meta
    Bases: object

    Forward metadata.

    handler_class = 'aries_cloudagent.protocols.routing.handlers.forward_handler.ForwardHandler'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/routing/1.0/forward'
    schema_class = 'ForwardSchema'
```

```
class aries_cloudagent.protocols.routing.messages.forward.ForwardSchema(*args,
                                                                           **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

Forward message schema used in serialization/deserialization.

```
class Meta
    Bases: object

    ForwardSchema metadata.

    model_class
        alias of Forward

    handle_str_message (data, **kwargs)
        Accept string value for msg, as produced by previous implementation.

    msg = <fields.Dict (default=<marshmallow.missing>, attribute=None, validate=None, required=None)>
    to = <fields.String (default=<marshmallow.missing>, attribute=None, validate=None, required=None)>
```

aries_cloudagent.protocols.routing.messages.route_query_request module

Query existing forwarding routes.

```
class aries_cloudagent.protocols.routing.messages.route_query_request.RouteQueryRequest (*,
                                             filter: Optional[Dict[str, Any]] = None,
                                             paginate: bool = False,
                                             **kwargs)

Bases: aries_cloudagent.messaging.agent_message.AgentMessage
```

Query existing routes from a routing agent.

```
class Meta
    Bases: object
    RouteQueryRequest metadata.
    handler_class = 'aries_cloudagent.protocols.routing.handlers.route_query_request_handler'
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/routing/1.0/route-query-request'
    schema_class = 'RouteQueryRequestSchema'
```

```
class aries_cloudagent.protocols.routing.messages.route_query_request.RouteQueryRequestSchema
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

RouteQueryRequest message schema used in serialization/deserialization.

```
class Meta
    Bases: object
    RouteQueryRequestSchema metadata.
    model_class
        alias of RouteQueryRequest
    filter = <fields.Dict (default=<marshmallow.missing>, attribute=None, validate=None, re
    paginate = <fields.Nested (default=<marshmallow.missing>, attribute=None, validate=None
```

aries_cloudagent.protocols.routing.messages.route_query_response module

Return existing forwarding routes in response to a query.

```
class aries_cloudagent.protocols.routing.messages.route_query_response.RouteQueryResponse (
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Return existing routes from a routing agent.

```
class Meta
```

Bases: *object*

RouteQueryResponse metadata.

```
    handler_class = 'aries_cloudagent.protocols.routing.handlers.route_query_response_h
```

```
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/routing/1.0/route-query-respons
```

```
    schema_class = 'RouteQueryResponseSchema'
```

```
class aries_cloudagent.protocols.routing.messages.route_query_response.RouteQueryResponseS
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

RouteQueryResponse message schema used in serialization/deserialization.

```
class Meta
```

Bases: *object*

RouteQueryResponseSchema metadata.

```
    model_class
```

alias of *RouteQueryResponse*

```
    paginated = <fields.Nested(default=<marshmallow.missing>, attribute=None, validate=None
```

```
    routes = <fields.List(default=<marshmallow.missing>, attribute=None, validate=None, re
```

`aries_cloudagent.protocols.routing.messages.route_update_request` module

Request to update forwarding routes.

```
class aries_cloudagent.protocols.routing.messages.route_update_request.RouteUpdateRequest (
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Request to existing routes with a routing agent.

```
class Meta
    Bases: object
    RouteUpdateRequest metadata.
    handler_class = 'aries_cloudagent.protocols.routing.handlers.route_update_request_h
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/routing/1.0/route-update-reques
    schema_class = 'RouteUpdateRequestSchema'

class aries_cloudagent.protocols.routing.messages.route_update_request.RouteUpdateRequestS

    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    RouteUpdateRequest message schema used in serialization/deserialization.

class Meta
    Bases: object
    RouteUpdateRequestSchema metadata.
    model_class
        alias of RouteUpdateRequest
    updates = <fields.List (default=<marshmallow.missing>, attribute=None, validate=None, r
```

aries_cloudagent.protocols.routing.messages.route_update_response module

Response for a route update request.

```
class aries_cloudagent.protocols.routing.messages.route_update_response.RouteUpdateResponse
```

```
    Bases: aries_cloudagent.messaging.agent_message.AgentMessage
    Response for a route update request.

class Meta
    Bases: object
    RouteUpdateResponse metadata.
    handler_class = 'aries_cloudagent.protocols.routing.handlers.route_update_response_
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/routing/1.0/route-update-respon
    schema_class = 'RouteUpdateResponseSchema'

class aries_cloudagent.protocols.routing.messages.route_update_response.RouteUpdateResponse

    Bases: aries_cloudagent.messaging.agent_message.AgentMessageSchema
    RouteUpdateResponse message schema used in serialization/deserialization.

class Meta
    Bases: object
    RouteUpdateResponseSchema metadata.
```



```

model_class
    alias of RouteUpdateResponse

```

```

updated = <fields.List (default=<marshmallow.missing>, attribute=None, validate=None, r

```

aries_cloudagent.protocols.routing.models package

Submodules

aries_cloudagent.protocols.routing.models.paginate module

An object for containing the request pagination information.

```

class aries_cloudagent.protocols.routing.models.paginate.Paginate(*, limit: int
                                = None,
                                offset: int
                                = None,
                                **kwargs)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing the pagination details of a request.

```

class Meta
    Bases: object
    Paginate metadata.
    schema_class = 'PaginateSchema'

```

```

class aries_cloudagent.protocols.routing.models.paginate.PaginateSchema(*args,
                                **kwargs)

```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Paginate schema.

```

class Meta
    Bases: object
    PaginateSchema metadata.
    model_class = 'Paginate'

```

```

limit = <fields.Integer (default=<marshmallow.missing>, attribute=None, validate=None,

```

```

offset = <fields.Integer (default=<marshmallow.missing>, attribute=None, validate=None,

```

aries_cloudagent.protocols.routing.models.paginated module

An object for containing the response pagination information.

```
class aries_cloudagent.protocols.routing.models.paginated.Paginated(*, start:
                                                                    int =
                                                                    None,
                                                                    end: int
                                                                    = None,
                                                                    limit: int
                                                                    = None,
                                                                    total: int
                                                                    = None,
                                                                    **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing the pagination details of a response.

```
class Meta
    Bases: object

    Paginated metadata.

    schema_class = 'PaginatedSchema'
```

```
class aries_cloudagent.protocols.routing.models.paginated.PaginatedSchema(*args,
                                                                    **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

Paginated schema.

```
class Meta
    Bases: object

    PaginatedSchema metadata.

    model_class = 'Paginated'
```

```
end = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=None, re
limit = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=None,
start = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=None,
total = <fields.Integer(default=<marshmallow.missing>, attribute=None, validate=None,
```

aries_cloudagent.protocols.routing.models.route_query_result module

An object for containing returned route information.

```
class aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResult(*,
                                                                                      re-
                                                                                      cip-
                                                                                      i-
                                                                                      ent_key:
                                                                                      str
                                                                                      =
                                                                                      None,
                                                                                      **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing route information returned by a route query.

```
class Meta
    Bases: object
```

RouteQueryResult metadata.

```
schema_class = 'RouteQueryResultSchema'
```

```
class aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResultSchema (
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

RouteQueryResult schema.

```
class Meta
```

Bases: *object*

RouteQueryResultSchema metadata.

```
model_class = 'RouteQueryResult'
```

```
recipient_key = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
```

aries_cloudagent.protocols.routing.models.route_record module

An object for containing information on an individual route.

```
class aries_cloudagent.protocols.routing.models.route_record.RouteRecord (*,
                                                                           record_id:
                                                                           str
                                                                           =
                                                                           None,
                                                                           con-
                                                                           nec-
                                                                           tion_id:
                                                                           str
                                                                           =
                                                                           None,
                                                                           re-
                                                                           cip-
                                                                           i-
                                                                           ent_key:
                                                                           str
                                                                           =
                                                                           None,
                                                                           cre-
                                                                           ated_at:
                                                                           str
                                                                           =
                                                                           None,
                                                                           up-
                                                                           dated_at:
                                                                           str
                                                                           =
                                                                           None,
                                                                           **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing stored route information.

```
class Meta
```

Bases: *object*

```
RouteRecord metadata.

    schema_class = 'RouteRecordSchema'

class aries_cloudagent.protocols.routing.models.route_record.RouteRecordSchema(*args,
                                                                                **kwargs)
    Bases: aries_cloudagent.messaging.models.base.BaseModelSchema
    RouteRecord schema.

    class Meta
        Bases: object
        RouteRecordSchema metadata.

        model_class = 'RouteRecord'

    connection_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    created_at = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    recipient_key = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    record_id = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
    updated_at = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None)>
```

aries_cloudagent.protocols.routing.models.route_update module

An object for containing route information to be updated.

```
class aries_cloudagent.protocols.routing.models.route_update.RouteUpdate(*,
                                recipient_key:
                                str
                                =
                                None,
                                action:
                                str
                                =
                                None,
                                **kwargs)

    Bases: aries_cloudagent.messaging.models.base.BaseModel

    Class representing a route update request.

    ACTION_CREATE = 'create'
    ACTION_DELETE = 'delete'

    class Meta
        Bases: object
        RouteUpdate metadata.

        schema_class = 'RouteUpdateSchema'

class aries_cloudagent.protocols.routing.models.route_update.RouteUpdateSchema(*args,
                                                                                **kwargs)
    Bases: aries_cloudagent.messaging.models.base.BaseModelSchema
```

RouteUpdate schema.

```
class Meta
    Bases: object

    RouteUpdateSchema metadata.

    model_class = 'RouteUpdate'

    action = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    recipient_key = <fields.String(default=<marshmallow.missing>, attribute=None, validate=
```

aries_cloudagent.protocols.routing.models.route_updated module

An object for containing updated route information.

```
class aries_cloudagent.protocols.routing.models.route_updated.RouteUpdated(*,
                                                                            re-
                                                                            cip-
                                                                            i-
                                                                            ent_key:
                                                                            str
                                                                            =
                                                                            None,
                                                                            ac-
                                                                            tion:
                                                                            str
                                                                            =
                                                                            None,
                                                                            re-
                                                                            sult:
                                                                            str
                                                                            =
                                                                            None,
                                                                            **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModel*

Class representing a route update response.

```
class Meta
    Bases: object

    RouteUpdated metadata.

    schema_class = 'RouteUpdatedSchema'

    RESULT_CLIENT_ERROR = 'client_error'

    RESULT_NO_CHANGE = 'no_change'

    RESULT_SERVER_ERROR = 'server_error'

    RESULT_SUCCESS = 'success'
```

```
class aries_cloudagent.protocols.routing.models.route_updated.RouteUpdatedSchema(*args,
                                                                                    **kwargs)
```

Bases: *aries_cloudagent.messaging.models.base.BaseModelSchema*

RouteUpdated schema.

```
class Meta
    Bases: object
    RouteUpdatedSchema metadata.
    model_class = 'RouteUpdated'
    action = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    recipient_key = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    result = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None, >
```

Submodules

aries_cloudagent.protocols.routing.manager module

Routing manager classes for tracking and inspecting routing records.

```
exception aries_cloudagent.protocols.routing.manager.RouteNotFoundError(*args,
                                                                    er-
                                                                    ror_code:
                                                                    str
                                                                    =
                                                                    None,
                                                                    **kwargs)

Bases: aries_cloudagent.protocols.routing.manager.RoutingManagerError

Requested route was not found.
```

```
class aries_cloudagent.protocols.routing.manager.RoutingManager(context:
                                                                    aries_cloudagent.config.injection_cont
```

Bases: object

Class for handling routing records.

RECORD_TYPE = 'forward_route'

context

Accessor for the current request context.

Returns The request context for this connection

create_route_record(client_connection_id: str = None, recipient_key: str = None) →
aries_cloudagent.protocols.routing.models.route_record.RouteRecord
Create and store a new RouteRecord.

Parameters

- **client_connection_id** – The ID of the connection record
- **recipient_key** – The recipient verkey of the route

Returns The new routing record

delete_route_record(route: aries_cloudagent.protocols.routing.models.route_record.RouteRecord)
Remove an existing route record.

get_recipient(recip_verkey: str) → aries_cloudagent.protocols.routing.models.route_record.RouteRecord
Resolve the recipient for a verkey.

Parameters **recip_verkey** – The verkey (“to”) of the incoming Forward message

Returns The *RouteRecord* associated with this verkey

get_routes (*client_connection_id*: *str* = *None*, *tag_filter*: *dict* = *None*) → *Sequence*[*aries_cloudagent.protocols.routing.models.route_record.RouteRecord*]
Fetch all routes associated with the current connection.

Parameters

- **client_connection_id** – The ID of the connection record
- **tag_filter** – An optional dictionary of tag filters

Returns A sequence of route records found by the query

send_create_route (*router_connection_id*: *str*, *recip_key*: *str*, *outbound_handler*)
Create and send a route update request.

Returns: the current routing state (request or done)

update_routes (*client_connection_id*: *str*, *updates*: *Sequence*[*aries_cloudagent.protocols.routing.models.route_update.RouteUpdate*]
→ *Sequence*[*aries_cloudagent.protocols.routing.models.route_updated.RouteUpdated*])
Update routes associated with the current connection.

Parameters

- **client_connection_id** – The ID of the connection record
- **updates** – The sequence of route updates (create/delete) to perform.

exception *aries_cloudagent.protocols.routing.manager.RoutingManagerError* (*args,
er-
ror_code:
str
=
None,
**kwargs)

Bases: *aries_cloudagent.core.error.BaseError*

Generic routing error.

aries_cloudagent.protocols.routing.message_types module

Message type identifiers for Routing.

3.1.12 aries_cloudagent.protocols.trustping package

Subpackages

aries_cloudagent.protocols.trustping.handlers package

Submodules

aries_cloudagent.protocols.trustping.handlers.ping_handler module

Ping handler.

class *aries_cloudagent.protocols.trustping.handlers.ping_handler.PingHandler*
Bases: *aries_cloudagent.messaging.base_handler.BaseHandler*

Ping handler class.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
 Handle ping message.

Parameters

- **context** – Request context
- **responder** – Responder used to reply

`aries_cloudagent.protocols.trustping.handlers.ping_response_handler` module

Ping response handler.

class `aries_cloudagent.protocols.trustping.handlers.ping_response_handler.PingResponseHandler`
 Bases: `aries_cloudagent.messaging.base_handler.BaseHandler`

Ping response handler class.

handle (*context:* `aries_cloudagent.messaging.request_context.RequestContext`, *responder:* `aries_cloudagent.messaging.responder.BaseResponder`)
 Handle ping response message.

Parameters

- **context** – Request context
- **responder** – Responder used to reply

`aries_cloudagent.protocols.trustping.messages` package

Submodules

`aries_cloudagent.protocols.trustping.messages.ping` module

Represents a trust ping message.

class `aries_cloudagent.protocols.trustping.messages.ping.Ping` (*, *re-*
ponse_requested:
bool = True, com-
ment: str = None,
***kwargs*)

Bases: `aries_cloudagent.messaging.agent_message.AgentMessage`

Class representing a trustping message.

class Meta

Bases: `object`

Ping metadata.

handler_class = `'aries_cloudagent.protocols.trustping.handlers.ping_handler.PingHandler'`

message_type = `'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/trust_ping/1.0/ping'`

schema_class = `'PingSchema'`

class `aries_cloudagent.protocols.trustping.messages.ping.PingSchema` (*args,
***kwargs*)

Bases: `aries_cloudagent.messaging.agent_message.AgentMessageSchema`

Schema for Ping class.

```
class Meta
    Bases: object

    PingSchema metadata.

    model_class
        alias of Ping

    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
    response_requested = <fields.Boolean(default=True, attribute=None, validate=None, requ
```

aries_cloudagent.protocols.trustping.messages.ping_response module

Represents an response to a trust ping message.

```
class aries_cloudagent.protocols.trustping.messages.ping_response.PingResponse(*,
                                                                              com-
                                                                              ment:
                                                                              str
                                                                              =
                                                                              None,
                                                                              **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessage*

Class representing a ping response.

```
class Meta
    Bases: object

    PingResponse metadata.

    handler_class = 'aries_cloudagent.protocols.trustping.handlers.ping_response_handle
    message_type = 'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec/trust_ping/1.0/ping_response'
    schema_class = 'PingResponseSchema'
```

```
class aries_cloudagent.protocols.trustping.messages.ping_response.PingResponseSchema(*args,
                                                                              **kwargs)
```

Bases: *aries_cloudagent.messaging.agent_message.AgentMessageSchema*

PingResponse schema.

```
class Meta
    Bases: object

    PingResponseSchema metadata.

    model_class
        alias of PingResponse

    comment = <fields.String(default=<marshmallow.missing>, attribute=None, validate=None,
```

Submodules

aries_cloudagent.protocols.trustping.message_types module

Message type identifiers for Trust Pings.

aries_cloudagent.protocols.trustping.routes module

Trust ping admin routes.

```
class aries_cloudagent.protocols.trustping.routes.PingRequestResponseSchema (*,  
                                                                    only=None,  
                                                                    ex-  
                                                                    clude=(),  
                                                                    many=False,  
                                                                    con-  
                                                                    text=None,  
                                                                    load_only=(),  
                                                                    dump_only=(),  
                                                                    par-  
                                                                    tial=False,  
                                                                    un-  
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Request schema for performing a ping.

opts = `<marshmallow.schema.SchemaOpts object>`

```
class aries_cloudagent.protocols.trustping.routes.PingRequestSchema (*,  
                                                                    only=None,  
                                                                    ex-  
                                                                    clude=(),  
                                                                    many=False,  
                                                                    con-  
                                                                    text=None,  
                                                                    load_only=(),  
                                                                    dump_only=(),  
                                                                    par-  
                                                                    tial=False,  
                                                                    un-  
                                                                    known=None)
```

Bases: `marshmallow.schema.Schema`

Request schema for performing a ping.

opts = `<marshmallow.schema.SchemaOpts object>`

```
aries_cloudagent.protocols.trustping.routes.connections_send_ping(request:  
                                                                    <sphinx.ext.autodoc.importer._Mock  
                                                                    object at  
                                                                    0x7f9e7590f358>)
```

Request handler for sending a trust ping to a connection.

Parameters **request** – aiohttp request object

```
aries_cloudagent.protocols.trustping.routes.register(app:  
                                                                    <sphinx.ext.autodoc.importer._MockObject  
                                                                    object at 0x7f9e7590f358>)
```

Register routes.

CHAPTER 4

Indices and tables

- `genindex`

a

aries_cloudagent, 3
aries_cloudagent.admin, 3
aries_cloudagent.admin.base_server, 3
aries_cloudagent.admin.error, 4
aries_cloudagent.admin.server, 4
aries_cloudagent.cache, 6
aries_cloudagent.cache.base, 6
aries_cloudagent.cache.basic, 7
aries_cloudagent.commands, 7
aries_cloudagent.commands.help, 8
aries_cloudagent.commands.provision, 8
aries_cloudagent.commands.start, 8
aries_cloudagent.config, 9
aries_cloudagent.config.argparse, 9
aries_cloudagent.config.base, 11
aries_cloudagent.config.base_context, 12
aries_cloudagent.config.default_context, 13
aries_cloudagent.config.error, 13
aries_cloudagent.config.injection_context, 13
aries_cloudagent.config.injector, 15
aries_cloudagent.config.ledger, 15
aries_cloudagent.config.logging, 15
aries_cloudagent.config.provider, 16
aries_cloudagent.config.settings, 17
aries_cloudagent.config.util, 18
aries_cloudagent.config.wallet, 18
aries_cloudagent.connections, 115
aries_cloudagent.connections.models, 115
aries_cloudagent.connections.models.connection_record, 124
aries_cloudagent.connections.models.connection_target, 129
aries_cloudagent.connections.models.diddoc, 115
aries_cloudagent.connections.models.diddoc.didoc, 118
aries_cloudagent.connections.models.diddoc.publickey, 120
aries_cloudagent.connections.models.diddoc.service, 122
aries_cloudagent.connections.models.diddoc.util, 123
aries_cloudagent.core, 18
aries_cloudagent.core.conductor, 18
aries_cloudagent.core.dispatcher, 20
aries_cloudagent.core.error, 21
aries_cloudagent.core.plugin_registry, 22
aries_cloudagent.core.protocol_registry, 22
aries_cloudagent.holder, 23
aries_cloudagent.holder.base, 23
aries_cloudagent.holder.indy, 23
aries_cloudagent.issuer, 25
aries_cloudagent.issuer.base, 25
aries_cloudagent.issuer.indy, 25
aries_cloudagent.ledger, 25
aries_cloudagent.ledger.base, 26
aries_cloudagent.ledger.error, 27
aries_cloudagent.ledger.indy, 27
aries_cloudagent.ledger.provider, 30
aries_cloudagent.ledger.routes, 30
aries_cloudagent.ledger.util, 32
aries_cloudagent.messaging, 32
aries_cloudagent.messaging.ack, 32
aries_cloudagent.messaging.ack.message, 32
aries_cloudagent.messaging.agent_message, 55
aries_cloudagent.messaging.base_handler, 58
aries_cloudagent.messaging.credential_definitions, 32
aries_cloudagent.messaging.credential_definitions.ledger, 32

aries_cloudagent.messaging.credential_definitions, 35	aries_cloudagent.protocols.actionmenu.handlers, per 132
aries_cloudagent.messaging.decorators, 35	aries_cloudagent.protocols.actionmenu.message_types, 139
aries_cloudagent.messaging.decorators.attach_decorator, 35	aries_cloudagent.protocols.actionmenu.messages, 132
aries_cloudagent.messaging.decorators.base, 39	aries_cloudagent.protocols.actionmenu.messages.menu, 132
aries_cloudagent.messaging.decorators.definition, 40	aries_cloudagent.protocols.actionmenu.messages.menu, 133
aries_cloudagent.messaging.decorators.load_definition, 40	aries_cloudagent.protocols.actionmenu.messages.per 134
aries_cloudagent.messaging.decorators.placeholder, 41	aries_cloudagent.protocols.actionmenu.models, 134
aries_cloudagent.messaging.decorators.signature, 42	aries_cloudagent.protocols.actionmenu.models.menu_ 134
aries_cloudagent.messaging.decorators.thread_decorator, 43	aries_cloudagent.protocols.actionmenu.models.menu_ 135
aries_cloudagent.messaging.decorators.timing_decorator, 44	aries_cloudagent.protocols.actionmenu.models.menu_ 137
aries_cloudagent.messaging.decorators.transport_decorator, 46	aries_cloudagent.protocols.actionmenu.routes, 139
aries_cloudagent.messaging.error, 58	aries_cloudagent.protocols.actionmenu.util, 141
aries_cloudagent.messaging.models, 47	
aries_cloudagent.messaging.models.base, 47	aries_cloudagent.protocols.basicmessage, 142
aries_cloudagent.messaging.models.base_record, 49	aries_cloudagent.protocols.basicmessage.handlers, 142
aries_cloudagent.messaging.request_context, 58	aries_cloudagent.protocols.basicmessage.handlers.ba 142
aries_cloudagent.messaging.responder, 59	aries_cloudagent.protocols.basicmessage.message_type 143
aries_cloudagent.messaging.schemas, 52	aries_cloudagent.protocols.basicmessage.messages, 142
aries_cloudagent.messaging.schemas.routes, 52	aries_cloudagent.protocols.basicmessage.messages.ba 142
aries_cloudagent.messaging.schemas.util, 55	aries_cloudagent.protocols.basicmessage.routes, 143
aries_cloudagent.messaging.util, 61	
aries_cloudagent.messaging.valid, 62	aries_cloudagent.protocols.connections, 144
aries_cloudagent.protocols, 131	
aries_cloudagent.protocols.actionmenu, 131	aries_cloudagent.protocols.connections.handlers, 144
aries_cloudagent.protocols.actionmenu.base_service, 138	aries_cloudagent.protocols.connections.handlers.cor 144
aries_cloudagent.protocols.actionmenu.connector, 138	aries_cloudagent.protocols.connections.handlers.cor 144
aries_cloudagent.protocols.actionmenu.driver_service, 139	aries_cloudagent.protocols.connections.handlers.cor 145
aries_cloudagent.protocols.actionmenu.handlers, 131	aries_cloudagent.protocols.connections.manager, 151
aries_cloudagent.protocols.actionmenu.handlers.schema_handler, 131	aries_cloudagent.protocols.connections.message_type 156
aries_cloudagent.protocols.actionmenu.handlers.schema_query_handler, 131	aries_cloudagent.protocols.connections.messages, 145

aries_cloudagent.protocols.connections.messagescloudagent.protocol.basicdiscovery.message_types,	145	179
aries_cloudagent.protocols.connections.messagescloudagent.protocol.requests_discovery.messages,	147	177
aries_cloudagent.protocols.connections.messagescloudagent.protocol.spoisediscovery.messages.disc	148	177
aries_cloudagent.protocols.connections.messagescloudagent.protocol.problem_reportprotocols.discovery.messages.query	149	178
aries_cloudagent.protocols.connections.messagescloudagent.protocol.routes,	150	179
aries_cloudagent.protocols.connections.messagescloudagent.protocol.introduction,	150	179
aries_cloudagent.protocols.connections.messagescloudagent.protocol.introduction.base_servi	156	183
aries_cloudagent.protocols.credentials, aries_cloudagent.protocols.introduction.demo_servi	159	184
aries_cloudagent.protocols.credentials.handlerscloudagent.protocol.introduction.handlers,	159	179
aries_cloudagent.protocols.credentials.handlerscloudagent.protocol.issue_handlerintroduction.handlers.fo	159	180
aries_cloudagent.protocols.credentials.handlerscloudagent.protocol.offer_handlerintroduction.handlers.in	160	180
aries_cloudagent.protocols.credentials.handlerscloudagent.protocol.requests_handlerintroduction.handlers.in	160	180
aries_cloudagent.protocols.credentials.handlerscloudagent.protocol.stored_handlerintroduction.message_type	160	185
aries_cloudagent.protocols.credentials.managercloudagent.protocol.introduction.messages,	167	180
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.introduction.messages.fo	169	180
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.introduction.messages.in	161	181
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.issue.introduction.messages.in	161	182
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.offer.introduction.routes,	161	185
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.requests.issue_credential,	162	185
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.stored.issue_credential.message	163	207
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.issue_credential.routes,	164	207
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.issue_credential.v1_0,	164	185
aries_cloudagent.protocols.credentials.messagescloudagent.protocol.issue_credential.v1_0.ha	169	185
aries_cloudagent.protocols.discovery, aries_cloudagent.protocol.issue_credential.v1_0.ha	176	185
aries_cloudagent.protocols.discovery.handlerscloudagent.protocol.issue_credential.v1_0.ha	176	186
aries_cloudagent.protocols.discovery.handlerscloudagent.protocol.issue_credential.v1_0.ha	176	186
aries_cloudagent.protocols.discovery.handlerscloudagent.protocol.issue_credential.v1_0.ha	177	186

`aries_cloudagent.utils.http`, 91
`aries_cloudagent.utils.repeat`, 91
`aries_cloudagent.utils.stats`, 92
`aries_cloudagent.utils.task_queue`, 93
`aries_cloudagent.verifier`, 95
`aries_cloudagent.verifier.base`, 95
`aries_cloudagent.verifier.indy`, 95
`aries_cloudagent.version`, 113
`aries_cloudagent.wallet`, 96
`aries_cloudagent.wallet.base`, 96
`aries_cloudagent.wallet.basic`, 99
`aries_cloudagent.wallet.crypto`, 102
`aries_cloudagent.wallet.error`, 105
`aries_cloudagent.wallet.indy`, 105
`aries_cloudagent.wallet.plugin`, 109
`aries_cloudagent.wallet.provider`, 110
`aries_cloudagent.wallet.routes`, 110
`aries_cloudagent.wallet.util`, 112

A

accept (*aries_cloudagent.connections.models.connection_record.ConnectionRecord* attribute), 128
 ACCEPT_AUTO (*aries_cloudagent.connections.models.connection_record.ConnectionRecord* attribute), 126
 ACCEPT_MANUAL (*aries_cloudagent.connections.models.connection_record.ConnectionRecord* attribute), 126
 accept_response () (*aries_cloudagent.protocols.connections.manager.ConnectionManager* method), 151
 accept_response () (*aries_cloudagent.transport.inbound.session.InboundSession* method), 78
 accept_taa () (in module *aries_cloudagent.config.ledger*), 15
 accept_txn_author_agreement () (*aries_cloudagent.ledger.base.BaseLedger* method), 26
 accept_txn_author_agreement () (*aries_cloudagent.ledger.indy.IndyLedger* method), 28
 AcceptResult (class in *aries_cloudagent.transport.inbound.session*), 77
 Ack (class in *aries_cloudagent.messaging.ack.message*), 32
 Ack.Meta (class in *aries_cloudagent.messaging.ack.message*), 32
 AckSchema (class in *aries_cloudagent.messaging.ack.message*), 32
 AckSchema.Meta (class in *aries_cloudagent.messaging.ack.message*), 32
 acquire () (*aries_cloudagent.cache.base.BaseCache* method), 6
 action (*aries_cloudagent.protocols.routing.models.route_update.RouteUpdateSchema* attribute), 249
 action (*aries_cloudagent.protocols.routing.models.route_update.RouteUpdateSchema* attribute), 250
 ACTION_CREATE (*aries_cloudagent.protocols.routing.models.route_update.RouteUpdateSchema* attribute), 248
 ACTION_DELETE (*aries_cloudagent.protocols.routing.models.route_update.RouteUpdateSchema* attribute), 248
 actionmenu_close () (in module *aries_cloudagent.protocols.actionmenu.routes*), 140
 actionmenu_fetch () (in module *aries_cloudagent.protocols.actionmenu.routes*), 141
 actionmenu_perform () (in module *aries_cloudagent.protocols.actionmenu.routes*), 141
 actionmenu_request () (in module *aries_cloudagent.protocols.actionmenu.routes*), 141
 actionmenu_send () (in module *aries_cloudagent.protocols.actionmenu.routes*), 141
 add_active () (*aries_cloudagent.utils.task_queue.TaskQueue* method), 93
 add_arguments () (*aries_cloudagent.config.argparse.AdminGroup* method), 9
 add_arguments () (*aries_cloudagent.config.argparse.ArgumentGroup* method), 9
 add_arguments () (*aries_cloudagent.config.argparse.DebugGroup* method), 9
 add_arguments () (*aries_cloudagent.config.argparse.GeneralGroup* method), 9
 add_arguments () (*aries_cloudagent.config.argparse.LedgerGroup* method), 10
 add_arguments () (*aries_cloudagent.config.argparse.LoggingGroup* method), 10
 add_arguments () (*aries_cloudagent.config.argparse.ProtocolGroup* method), 10
 add_arguments () (*aries_cloudagent.config.argparse.TransportGroup* method), 10
 add_arguments () (*aries_cloudagent.config.argparse.WalletGroup* method), 11

[add_key_for_did\(\)](#) [57](#)
 (aries_cloudagent.protocols.connections.manager.ConnectionManagerSchema.Meta (class in
 method), [152](#) aries_cloudagent.messaging.agent_message),
[add_message\(\)](#) (aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue
 method), [72](#) alias (aries_cloudagent.connections.models.connection_record.ConnectionRecord),
[add_model\(\)](#) (aries_cloudagent.messaging.decorators.base.BaseDecoratorSet
 method), [39](#) AMLRecordSchema (class in
[add_pending\(\)](#) (aries_cloudagent.utils.task_queue.TaskQueue aries_cloudagent.ledger.routes), [30](#)
 method), [93](#) ArgsParseError, [13](#)
[add_record\(\)](#) (aries_cloudagent.storage.base.BaseStorageArgumentGroup (class in
 method), [63](#) aries_cloudagent.config.argparse), [9](#)
[add_record\(\)](#) (aries_cloudagent.storage.basic.BasicStorage aries_cloudagent (module), [3](#)
 method), [65](#) aries_cloudagent.admin (module), [3](#)
[add_record\(\)](#) (aries_cloudagent.storage.indy.IndyStorage aries_cloudagent.admin.base_server (mod-
 method), [68](#) ule), [3](#)
[add_reply_thread_ids\(\)](#) aries_cloudagent.admin.error (module), [4](#)
 (aries_cloudagent.transport.inbound.session.InboundSession aries_cloudagent.admin.server (module), [4](#)
 method), [78](#) aries_cloudagent.cache (module), [6](#)
[add_reply_verkeys\(\)](#) aries_cloudagent.cache.base (module), [6](#)
 (aries_cloudagent.transport.inbound.session.InboundSession aries_cloudagent.cache.basic (module), [7](#)
 method), [78](#) aries_cloudagent.commands (module), [7](#)
[add_service_pubkeys\(\)](#) aries_cloudagent.commands.help (module), [8](#)
 (aries_cloudagent.connections.models.diddoc.DIDDoc aries_cloudagent.commands.provision
 method), [115](#) (module), [8](#)
[add_service_pubkeys\(\)](#) aries_cloudagent.commands.start (module),
 (aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc aries_cloudagent.config (module), [9](#)
 method), [119](#) aries_cloudagent.config.argparse (mod-
[add_webhook_target\(\)](#) ule), [9](#)
 (aries_cloudagent.admin.base_server.BaseAdminServer aries_cloudagent.config.base (module), [11](#)
 method), [3](#) aries_cloudagent.config.base_context
[add_webhook_target\(\)](#) (aries_cloudagent.admin.server.AdminServer (module), [12](#)
 method), [4](#) aries_cloudagent.config.default_context
 AdminError, [4](#) (module), [13](#)
AdminGroup (class in aries_cloudagent.config.error (module), [13](#)
 aries_cloudagent.config.injection_context
AdminModulesSchema (class in (module), [13](#)
 aries_cloudagent.config.injector (mod-
AdminResponder (class in ule), [15](#)
 aries_cloudagent.config.ledger (module),
AdminServer (class in [15](#)
 aries_cloudagent.config.logging (module),
AdminSetupError, [4](#) [15](#)
AdminStatusSchema (class in aries_cloudagent.config.provider (mod-
 aries_cloudagent.admin.server), [5](#) ule), [16](#)
AgentMessage (class in aries_cloudagent.config.settings (mod-
 aries_cloudagent.messaging.agent_message), ule), [17](#)
[55](#) aries_cloudagent.config.util (module), [18](#)
AgentMessage.Meta (class in aries_cloudagent.config.wallet (module),
 aries_cloudagent.messaging.agent_message), [18](#)
[55](#) aries_cloudagent.connections (module), [115](#)
AgentMessageError, [56](#) aries_cloudagent.connections.models
AgentMessageSchema (class in (module), [115](#)
 aries_cloudagent.messaging.agent_message), aries_cloudagent.connections.models.connection_rec

(module), 124
 aries_cloudagent.connections.models.connector (module), 129
 aries_cloudagent.connections.models.did (module), 115
 aries_cloudagent.connections.models.did (module), 118
 aries_cloudagent.connections.models.did (module), 120
 aries_cloudagent.connections.models.did (module), 122
 aries_cloudagent.connections.models.did (module), 123
 aries_cloudagent.core (module), 18
 aries_cloudagent.core.conductor (module), 18
 aries_cloudagent.core.dispatcher (module), 20
 aries_cloudagent.core.error (module), 21
 aries_cloudagent.core.plugin_registry (module), 22
 aries_cloudagent.core.protocol_registry (module), 22
 aries_cloudagent.holder (module), 23
 aries_cloudagent.holder.base (module), 23
 aries_cloudagent.holder.indy (module), 23
 aries_cloudagent.issuer (module), 25
 aries_cloudagent.issuer.base (module), 25
 aries_cloudagent.issuer.indy (module), 25
 aries_cloudagent.ledger (module), 25
 aries_cloudagent.ledger.base (module), 26
 aries_cloudagent.ledger.error (module), 27
 aries_cloudagent.ledger.indy (module), 27
 aries_cloudagent.ledger.provider (module), 30
 aries_cloudagent.ledger.routes (module), 30
 aries_cloudagent.ledger.util (module), 32
 aries_cloudagent.messaging (module), 32
 aries_cloudagent.messaging.ack (module), 32
 aries_cloudagent.messaging.ack.message (module), 32
 aries_cloudagent.messaging.agent_message (module), 55
 aries_cloudagent.messaging.base_handler (module), 58
 aries_cloudagent.messaging.credential_definition (module), 32
 aries_cloudagent.messaging.credential_definition (module), 32
 aries_cloudagent.messaging.credential_definition (module), 32
 aries_cloudagent.messaging.credential_definition (module), 35
 aries_cloudagent.messaging.decorators (module), 35
 aries_cloudagent.messaging.decorators.attach_decorator (module), 35
 aries_cloudagent.messaging.decorators.base (module), 39
 aries_cloudagent.messaging.decorators.default (module), 40
 aries_cloudagent.messaging.decorators.localization (module), 40
 aries_cloudagent.messaging.decorators.please_ack_decorator (module), 41
 aries_cloudagent.messaging.decorators.signature_decorator (module), 42
 aries_cloudagent.messaging.decorators.thread_decorator (module), 43
 aries_cloudagent.messaging.decorators.timing_decorator (module), 44
 aries_cloudagent.messaging.decorators.transport_decorator (module), 46
 aries_cloudagent.messaging.error (module), 58
 aries_cloudagent.messaging.models (module), 47
 aries_cloudagent.messaging.models.base (module), 47
 aries_cloudagent.messaging.models.base_record (module), 49
 aries_cloudagent.messaging.request_context (module), 58
 aries_cloudagent.messaging.responder (module), 59
 aries_cloudagent.messaging.schemas (module), 52
 aries_cloudagent.messaging.schemas.routes (module), 52
 aries_cloudagent.messaging.schemas.util (module), 55
 aries_cloudagent.messaging.util (module), 61
 aries_cloudagent.messaging.valid (module), 62
 aries_cloudagent.protocols (module), 131
 aries_cloudagent.protocols.actionmenu (module), 131
 aries_cloudagent.protocols.actionmenu.base_service (module), 138
 aries_cloudagent.protocols.actionmenu.controller (module), 138
 aries_cloudagent.protocols.actionmenu.driver_service (module), 139
 aries_cloudagent.protocols.actionmenu.handlers (module), 131
 aries_cloudagent.protocols.actionmenu.handlers.menu (module), 131

```

aries_cloudagent.protocols.actionmenu.handlers.schemadagentprotocolhandlerconnections.messages
    (module), 131
    (module), 145
aries_cloudagent.protocols.actionmenu.handlers.schemadagentprotocolhandlerconnections.messages.cor
    (module), 132
    (module), 145
aries_cloudagent.protocols.actionmenu.messages_typeadagent.protocols.connections.messages.cor
    (module), 139
    (module), 147
aries_cloudagent.protocols.actionmenu.messages_scloudagent.protocols.connections.messages.cor
    (module), 132
    (module), 148
aries_cloudagent.protocols.actionmenu.messages_scloudagent.protocols.connections.messages.pr
    (module), 132
    (module), 149
aries_cloudagent.protocols.actionmenu.messages_scloudagent.protocols.connections.models
    (module), 133
    (module), 150
aries_cloudagent.protocols.actionmenu.messages_scloudagent.protocols.connections.models.conne
    (module), 134
    (module), 150
aries_cloudagent.protocols.actionmenu.models_aries_cloudagent.protocols.connections.routes
    (module), 134
    (module), 156
aries_cloudagent.protocols.actionmenu.models_scloudagent.protocols.credentials
    (module), 134
    (module), 159
aries_cloudagent.protocols.actionmenu.models_scloudagent.protocols.credentials.handlers
    (module), 135
    (module), 159
aries_cloudagent.protocols.actionmenu.models_scloudagent.protocols.credentials.handlers.cre
    (module), 137
    (module), 159
aries_cloudagent.protocols.actionmenu.routes_aries_cloudagent.protocols.credentials.handlers.cre
    (module), 139
    (module), 160
aries_cloudagent.protocols.actionmenu.utilities_aries_cloudagent.protocols.credentials.handlers.cre
    (module), 141
    (module), 160
aries_cloudagent.protocols.basicmessage_aries_cloudagent.protocols.credentials.handlers.cre
    (module), 142
    (module), 160
aries_cloudagent.protocols.basicmessage.handlers_scloudagent.protocols.credentials.manager
    (module), 142
    (module), 167
aries_cloudagent.protocols.basicmessage.handlers_scloudagent.protocols.credentials.message_type
    (module), 142
    (module), 169
aries_cloudagent.protocols.basicmessage.messages_typeadagent.protocols.credentials.messages
    (module), 143
    (module), 161
aries_cloudagent.protocols.basicmessage.messages_scloudagent.protocols.credentials.messages.cre
    (module), 142
    (module), 161
aries_cloudagent.protocols.basicmessage.messages_scloudagent.protocols.credentials.messages.cre
    (module), 142
    (module), 161
aries_cloudagent.protocols.basicmessage.routes_scloudagent.protocols.credentials.messages.cre
    (module), 143
    (module), 162
aries_cloudagent.protocols.connections_aries_cloudagent.protocols.credentials.messages.cre
    (module), 144
    (module), 163
aries_cloudagent.protocols.connections.handlers_scloudagent.protocols.credentials.models
    (module), 144
    (module), 164
aries_cloudagent.protocols.connections.handlers_scloudagent.protocolinvitationhandlers.models.crede
    (module), 144
    (module), 164
aries_cloudagent.protocols.connections.handlers_scloudagent.protocolrequests_handlers.credentials.routes
    (module), 144
    (module), 169
aries_cloudagent.protocols.connections.handlers_scloudagent.protocolresponses_handlervery
    (module), 145
    (module), 176
aries_cloudagent.protocols.connections.managercloudagent.protocols.discovery.handlers
    (module), 151
    (module), 176
aries_cloudagent.protocols.connections.messages_typeadagent.protocols.discovery.handlers.disc
    (module), 156
    (module), 176

```


[aries_cloudagent.protocols.discovery.handlers.query_handler \(module\)](#), 177
[aries_cloudagent.protocols.discovery.handlers.query_handler \(module\)](#), 186
[aries_cloudagent.protocols.discovery.messages.issue_credential.v1_0.handler \(module\)](#), 179
[aries_cloudagent.protocols.discovery.messages.issue_credential.v1_0.handler \(module\)](#), 187
[aries_cloudagent.protocols.discovery.messages.issue_credential.v1_0.handler \(module\)](#), 198
[aries_cloudagent.protocols.discovery.messages.issue_credential.v1_0.handler \(module\)](#), 201
[aries_cloudagent.protocols.discovery.messages.issue_credential.v1_0.handler \(module\)](#), 187
[aries_cloudagent.protocols.discovery.messages.issue_credential.v1_0.handler \(module\)](#), 189
[aries_cloudagent.protocols.introduction \(module\)](#), 179
[aries_cloudagent.protocols.introduction \(module\)](#), 189
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 183
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 191
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 192
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 193
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 187
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 187
[aries_cloudagent.protocols.introduction.handlers.issue_credential.v1_0.handler \(module\)](#), 195
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 185
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 195
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 201
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 207
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 228
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 228
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 207
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 207
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 208
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 208
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 219
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 222
[aries_cloudagent.protocols.introduction.messages.issue_credential.v1_0.handler \(module\)](#), 208

[illegible]

[aries_cloudagent.protocols.trustping.messages \(module\)](#), 252
[aries_cloudagent.protocols.trustping.messages_ping \(module\)](#), 252
[aries_cloudagent.protocols.trustping.messages_ping_response \(module\)](#), 253
[aries_cloudagent.protocols.trustping.routes \(module\)](#), 254
[aries_cloudagent.storage \(module\)](#), 63
[aries_cloudagent.storage.base \(module\)](#), 63
[aries_cloudagent.storage.basic \(module\)](#), 65
[aries_cloudagent.storage.error \(module\)](#), 67
[aries_cloudagent.storage.indy \(module\)](#), 68
[aries_cloudagent.storage.provider \(module\)](#), 70
[aries_cloudagent.storage.record \(module\)](#), 70
[aries_cloudagent.transport \(module\)](#), 71
[aries_cloudagent.transport.error \(module\)](#), 86
[aries_cloudagent.transport.inbound \(module\)](#), 71
[aries_cloudagent.transport.inbound.base \(module\)](#), 71
[aries_cloudagent.transport.inbound.deliver \(module\)](#), 72
[aries_cloudagent.transport.inbound.http \(module\)](#), 73
[aries_cloudagent.transport.inbound.manager \(module\)](#), 74
[aries_cloudagent.transport.inbound.message \(module\)](#), 75
[aries_cloudagent.transport.inbound.receive \(module\)](#), 76
[aries_cloudagent.transport.inbound.session \(module\)](#), 77
[aries_cloudagent.transport.inbound.ws \(module\)](#), 79
[aries_cloudagent.transport.outbound \(module\)](#), 80
[aries_cloudagent.transport.outbound.base \(module\)](#), 80
[aries_cloudagent.transport.outbound.http \(module\)](#), 81
[aries_cloudagent.transport.outbound.manager \(module\)](#), 81
[aries_cloudagent.transport.outbound.message \(module\)](#), 83
[aries_cloudagent.transport.outbound.ws \(module\)](#), 84
[aries_cloudagent.transport.pack_format \(module\)](#), 87
[aries_cloudagent.transport.queue \(module\)](#), 85
[aries_cloudagent.transport.queue.base \(module\)](#), 85
[aries_cloudagent.transport.queue.basic \(module\)](#), 85
[aries_cloudagent.transport.stats \(module\)](#), 87
[aries_cloudagent.transport.wire_format \(module\)](#), 88
[aries_cloudagent.utils \(module\)](#), 89
[aries_cloudagent.utils.classloader \(module\)](#), 89
[aries_cloudagent.utils.http \(module\)](#), 91
[aries_cloudagent.utils.repeat \(module\)](#), 91
[aries_cloudagent.utils.stats \(module\)](#), 92
[aries_cloudagent.utils.task_queue \(module\)](#), 93
[aries_cloudagent.verifier \(module\)](#), 95
[aries_cloudagent.verifier.base \(module\)](#), 95
[aries_cloudagent.verifier.indy \(module\)](#), 95
[aries_cloudagent.version \(module\)](#), 113
[aries_cloudagent.wallet \(module\)](#), 96
[aries_cloudagent.wallet.base \(module\)](#), 96
[aries_cloudagent.wallet.basic \(module\)](#), 99
[aries_cloudagent.wallet.crypto \(module\)](#), 102
[aries_cloudagent.wallet.error \(module\)](#), 105
[aries_cloudagent.wallet.indy \(module\)](#), 105
[aries_cloudagent.wallet.plugin \(module\)](#), 109
[aries_cloudagent.wallet.provider \(module\)](#), 110
[aries_cloudagent.wallet.routes \(module\)](#), 110
[aries_cloudagent.wallet.util \(module\)](#), 112
[assign_thread_from\(\)](#) ([aries_cloudagent.messaging.agent_message.AgentMessage](#) method), 55
[assign_thread_id\(\)](#) ([aries_cloudagent.messaging.agent_message.AgentMessage](#) method), 55
[attach_invitation\(\)](#) ([aries_cloudagent.connections.models.connection_record.ConnectionRecord](#) method), 126
[attach_request\(\)](#) ([aries_cloudagent.connections.models.connection_record.ConnectionRecord](#) method), 127
[AttachDecorator](#) (class in [aries_cloudagent.messaging.decorators.attach_decorator](#)), 35
[AttachDecorator.Meta](#) (class in

`aries_cloudagent.messaging.decorators.attach_decorator`), `aries_cloudagent.commands`), 7
36
`AttachDecoratorData` (class in **B**
`aries_cloudagent.messaging.decorators.attach_decorator`), `b64_to_bytes()` (in module
37 `aries_cloudagent.wallet.util`), 112
`AttachDecoratorData.Meta` (class in `b64_decoded_value()`
`aries_cloudagent.messaging.decorators.attach_decorator`), (`aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.p`
37 `method`), 187
`AttachDecoratorDataSchema` (class in `b64_decoded_value()`
`aries_cloudagent.messaging.decorators.attach_decorator`), (`aries_cloudagent.protocols.present_proof.v1_0.messages.inner.p`
38 `method`), 209
`AttachDecoratorDataSchema.Meta` (class in `b64_to_bytes()` (in module
`aries_cloudagent.messaging.decorators.attach_decorator`), `aries_cloudagent.wallet.util`), 112
38
`AttachDecoratorSchema` (class in `b64_to_str()` (in module
`aries_cloudagent.wallet.util`), 112
38 `aries_cloudagent.messaging.decorators.attach_decorator`), `BadRequestError`, 27
`AttachDecoratorSchema.Meta` (class in `attribute`), 37
38 `aries_cloudagent.messaging.decorators.attach_decorator`), (`class in aries_cloudagent.messaging.valid`), 62
`attr_dict()` (`aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.CredentialPreview`
`method`), 188
`Base64URL` (class in
`aries_cloudagent.messaging.valid`), 62
`aries_cloudagent.protocols.issue_credential.v1_0.routes`, `BaseAdminServer` (class in
204 `aries_cloudagent.admin.base_server`), 3
`attributes(aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview.CredentialPreviewSchema`
`attribute`), 189
`attributes(aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_preview.PresentationPreviewSchema`
`attribute`), 212
`authn(aries_cloudagent.connections.models.diddoc.PublicKey`
`attribute`), 117
`authn(aries_cloudagent.connections.models.diddoc.publickey.PublicKey`
`attribute`), 121
`authn_type(aries_cloudagent.connections.models.diddoc.LinkedDataKeySpec`
`attribute`), 116
`authn_type(aries_cloudagent.connections.models.diddoc.publickey.LinkedDataKeySpec`
`attribute`), 120
`authn_type(aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType` (class in
`attribute`), 121
`aries_cloudagent.config.base`), 11
`authn_type(aries_cloudagent.connections.models.diddoc.PublicKeyType`
`attribute`), 117
`BaseIntroductionService` (class in
`aries_cloudagent.protocols.introduction.base_service`),
`authnkey(aries_cloudagent.connections.models.diddoc.DIDDoc` 183
`attribute`), 116
`authnkey(aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`
`attribute`), 119
`BaseLedger` (class in `aries_cloudagent.ledger.base`),
`auto_issue(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema`
`attribute`), 167
`auto_issue(aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchangeSchema`
`attribute`), 198
`auto_offer(aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchangeSchema`
`attribute`), 198
`auto_present(aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10PresentationExchangeSchema`
`attribute`), 218
`available_commands()` (in module `aries_cloudagent.messaging.models.base`),

47			BasicMessageHandler	(class	in	
BaseModel.Meta	(class	in	aries_cloudagent.protocols.basicmessage.handlers.basicmessage			
aries_cloudagent.messaging.models.base),			142			
47			BasicMessageQueue	(class	in	
BaseModelError, 48			aries_cloudagent.transport.queue.basic),			
BaseModelSchema	(class	in	85			
aries_cloudagent.messaging.models.base),			BasicMessageSchema	(class	in	
48			aries_cloudagent.protocols.basicmessage.messages.basicmessage			
BaseModelSchema.Meta	(class	in	143			
aries_cloudagent.messaging.models.base),			BasicMessageSchema.Meta	(class	in	
48			aries_cloudagent.protocols.basicmessage.messages.basicmessage			
BaseOutboundTransport	(class	in	143			
aries_cloudagent.transport.outbound.base),	80		BasicStorage	(class	in	
BaseProvider	(class	in	aries_cloudagent.storage.basic),	65		
aries_cloudagent.config.base),	11		BasicStorageRecordSearch	(class	in	
BaseRecord	(class	in	aries_cloudagent.storage.basic),	67		
aries_cloudagent.messaging.models.base_record)			BasicWallet	(class in	aries_cloudagent.wallet.basic),	
49			99			
BaseRecord.Meta	(class	in	bind_instance()	(aries_cloudagent.config.injector.Injector		
aries_cloudagent.messaging.models.base_record),			method),	15		
49			bind_provider()	(aries_cloudagent.config.injector.Injector		
BaseRecordSchema	(class	in	method),	15		
aries_cloudagent.messaging.models.base_record)			bind_providers()	(aries_cloudagent.config.default_context.DefaultContextBuilder		
52			method),	13		
BaseRecordSchema.Meta	(class	in	build()	(aries_cloudagent.config.base_context.ContextBuilder		
aries_cloudagent.messaging.models.base_record),			method),	13		
52			build()	(aries_cloudagent.config.default_context.DefaultContextBuilder		
BaseResponder	(class	in	method),	13		
aries_cloudagent.messaging.responder),			byte_count	(aries_cloudagent.messaging.decorators.attach_decorator.A		
59			attribute),	38		
BaseSettings	(class	in	bytes_to_b58()	(in	module	
aries_cloudagent.config.base),	11		aries_cloudagent.wallet.util),	112		
BaseStorage	(class	in	bytes_to_b64()	(in	module	
aries_cloudagent.storage.base),	63		aries_cloudagent.wallet.util),	112		
BaseStorageRecordSearch	(class	in	ByteSize	(class in	aries_cloudagent.config.util),	18
aries_cloudagent.storage.base),	64					
BaseVerifier	(class	in				
aries_cloudagent.verifier.base),	95					
BaseWallet	(class in	aries_cloudagent.wallet.base),				
96						
BaseWireFormat	(class	in				
aries_cloudagent.transport.wire_format),						
88						
basic_tag_query_match()	(in	module				
aries_cloudagent.storage.basic),	67					
basic_tag_value_match()	(in	module				
aries_cloudagent.storage.basic),	67					
BasicCache	(class in	aries_cloudagent.cache.basic),	7			
BasicMessage	(class	in				
aries_cloudagent.protocols.basicmessage.messages.basicmessage)						
142						
BasicMessage.Meta	(class	in				
aries_cloudagent.protocols.basicmessage.messages.basicmessage)						
143						

`cancel()` (*aries_cloudagent.utils.task_queue.TaskQueue* `clear_binding()` (*aries_cloudagent.config.injector.Injector* `method`), 94 `method`), 15
`cancel_pending()` (*aries_cloudagent.utils.task_queue.TaskQueue* `cancel_pending()` (*aries_cloudagent.messaging.models.base_record.BaseRecord* `method`), 94 `method`), 49
`cancelled()` (*aries_cloudagent.utils.task_queue.PendingTask* `clear_cached_key()` (*aries_cloudagent.messaging.models.base_record.BaseRecord* `attribute`), 93 `class method`), 49
`cancelled()` (*aries_cloudagent.utils.task_queue.TaskQueue* `clear_response()` (*aries_cloudagent.transport.inbound.session.InboundSession* `attribute`), 94 `method`), 78
`canon()` (*in module aries_cloudagent.messaging.util*), 61 `clear_value()` (*aries_cloudagent.config.settings.Settings* `method`), 17
`canon_id()` (*in module aries_cloudagent.connections.models.diddoc.util*), 123 `close()` (*aries_cloudagent.ledger.indy.IndyLedger* `method`), 28
`canon_ref()` (*in module aries_cloudagent.connections.models.diddoc.util*), 123 `close()` (*aries_cloudagent.storage.base.BaseStorageRecordSearch* `method`), 64
`catalogs()` (*aries_cloudagent.messaging.decorators.localization_decorator.localization_decorator* `attribute`), 41 `close()` (*aries_cloudagent.storage.basic.BasicStorageRecordSearch* `method`), 70
`CATEGORIES()` (*aries_cloudagent.config.argparse.AdminGroup* `attribute`), 9 `close()` (*aries_cloudagent.storage.indy.IndyStorageRecordSearch* `method`), 78
`CATEGORIES()` (*aries_cloudagent.config.argparse.DebugGroup* `attribute`), 9 `close()` (*aries_cloudagent.transport.inbound.session.InboundSession* `method`), 96
`CATEGORIES()` (*aries_cloudagent.config.argparse.GeneralGroup* `attribute`), 9 `close()` (*aries_cloudagent.wallet.base.BaseWallet* `method`), 99
`CATEGORIES()` (*aries_cloudagent.config.argparse.LedgerGroup* `attribute`), 10 `close()` (*aries_cloudagent.wallet.basic.BasicWallet* `method`), 106
`CATEGORIES()` (*aries_cloudagent.config.argparse.LoggingGroup* `attribute`), 10 `closed()` (*aries_cloudagent.transport.inbound.session.InboundSession* `attribute`), 78
`CATEGORIES()` (*aries_cloudagent.config.argparse.ProtocolGroup* `attribute`), 10 `closed_session()` (*aries_cloudagent.transport.inbound.manager.InboundManager* `method`), 74
`CATEGORIES()` (*aries_cloudagent.config.argparse.TransportGroup* `attribute`), 10 `ClosedPoolError`, 27
`CATEGORIES()` (*aries_cloudagent.config.argparse.WalletGroup* `attribute`), 10 `Collector` (*aries_cloudagent.transport.outbound.base.BaseOutboundTransport* `attribute`), 80
`check_dump_decorators()` (*aries_cloudagent.messaging.agent_message.AgentMessageSchema* `method`), 57 `Collector` (*class in aries_cloudagent.utils.stats*), 92
`check_existing_schema()` (*aries_cloudagent.ledger.indy.IndyLedger* `method`), 28 `comment` (*aries_cloudagent.protocols.credentials.messages.credential_request.CredentialRequest* `attribute`), 162
`check_pool_config()` (*aries_cloudagent.ledger.indy.IndyLedger* `method`), 28 `comment` (*aries_cloudagent.protocols.discovery.messages.query.QuerySchema* `attribute`), 178
`ClassLoader` (*class in aries_cloudagent.utils.classloader*), 89 `comment` (*aries_cloudagent.protocols.issue_credential.v1_0.messages.create_credential_v1_0.CreateCredentialV10* `attribute`), 190
`ClassNotFoundError`, 90 `comment` (*aries_cloudagent.protocols.issue_credential.v1_0.messages.create_credential_v1_0.CreateCredentialV10* `attribute`), 192
`ClassProvider` (*class in aries_cloudagent.config.provider*), 16 `comment` (*aries_cloudagent.protocols.issue_credential.v1_0.messages.create_credential_v1_0.CreateCredentialV10* `attribute`), 193
`ClassProvider.Inject` (*class in aries_cloudagent.config.provider*), 16 `comment` (*aries_cloudagent.protocols.issue_credential.v1_0.messages.create_credential_v1_0.CreateCredentialV10* `attribute`), 194
`clear()` (*aries_cloudagent.cache.base.BaseCache* `method`), 6 `comment` (*aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof_v1_0.PresentProofV10* `attribute`), 213
`clear()` (*aries_cloudagent.cache.basic.BasicCache* `method`), 7 `comment` (*aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof_v1_0.PresentProofV10* `attribute`), 215
`comment` (*aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof_v1_0.PresentProofV10* `attribute`), 215

`attribute`), 216
`comment (aries_cloudagent.protocols.presentations.messages.presentation.presentation.CredentialPresentationSchema`
`attribute)`, 229
`comment (aries_cloudagent.protocols.presentations.messages.presentation.presentation.PresentationRequestSchema`
`attribute)`, 230
`comment (aries_cloudagent.protocols.trustping.messages.ping.PingSchema (aries_cloudagent.protocols.connections.routes),`
`attribute)`, 253
`comment (aries_cloudagent.protocols.trustping.messages.ping.response.PingResponseSchema (class in`
`attribute)`, 253
`common_config () (in module`
`aries_cloudagent.config.util)`, 18
`complete () (aries_cloudagent.core.dispatcher.Dispatcher`
`method)`, 20
`complete () (aries_cloudagent.utils.task_queue.TaskQueue`
`method)`, 94
`completed_task () (aries_cloudagent.utils.task_queue.TaskQueue`
`method)`, 94
`CompletedTask (class in`
`aries_cloudagent.utils.task_queue)`, 93
`Conductor (class in aries_cloudagent.core.conductor)`, 18
`ConfigError`, 12
`configure () (aries_cloudagent.config.logging.LoggingConfigurator`
`class method)`, 16
`connection (aries_cloudagent.protocols.connections.messages.connection_request.ConnectionRequestSchema`
`attribute)`, 148
`connection (aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitationSchema`
`attribute)`, 149
`connection_id (aries_cloudagent.connections.models.connection_record.ConnectionRecord`
`attribute)`, 127
`connection_id (aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema`
`attribute)`, 128
`connection_id (aries_cloudagent.protocols.credentials.models.credentials.credentials.CredentialsSchema`
`attribute)`, 167
`connection_id (aries_cloudagent.protocols.issue_credentials.models.credentials.exchange.V10CredentialExchangeSchema`
`attribute)`, 198
`connection_id (aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_exchange.V10PresentationExchangeSchema`
`attribute)`, 218
`connection_id (aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchangeSchema`
`attribute)`, 232
`connection_id (aries_cloudagent.protocols.routing.models.route_record.RouteRecordSchema`
`attribute)`, 248
`connection_id (aries_cloudagent.transport.inbound.receipt.MessageReceipt (aries_cloudagent.connections.models.connection_record),`
`attribute)`, 76
`connection_queued_end ()`
`(aries_cloudagent.transport.stats.StatsTracer`
`method)`, 88
`connection_queued_start ()`
`(aries_cloudagent.transport.stats.StatsTracer`
`method)`, 88
`connection_ready (aries_cloudagent.messaging.request_context.RequestContextSchema.Meta (class in`
`attribute)`, 59
`connection_ready ()`
`(aries_cloudagent.transport.stats.StatsTracer`
`method)`, 88

<code>aries_cloudagent.protocols.connections.messages.connection_request()</code>	<code>aries_cloudagent.protocols.connections.routes)</code>
147	159
<code>ConnectionRequest.Meta</code> (class in <code>connections_send_message()</code> (in module <code>aries_cloudagent.protocols.connections.messages.connection_request()</code> (in module <code>aries_cloudagent.protocols.basicmessage.routes)</code>),	<code>aries_cloudagent.protocols.basicmessage.routes)</code>
147	143
<code>ConnectionRequestHandler</code> (class in <code>connections_send_ping()</code> (in module <code>aries_cloudagent.protocols.connections.handlers.connection_request_handler()</code> (in module <code>aries_cloudagent.protocols.trustping.routes)</code>),	<code>aries_cloudagent.protocols.trustping.routes)</code>
144	254
<code>ConnectionRequestSchema</code> (class in <code>ConnectionStaticRequestSchema</code> (class in <code>aries_cloudagent.protocols.connections.messages.connection_request()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>),	<code>aries_cloudagent.protocols.connections.routes)</code>
148	156
<code>ConnectionRequestSchema.Meta</code> (class in <code>ConnectionStaticResultSchema</code> (class in <code>aries_cloudagent.protocols.connections.messages.connection_request()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>),	<code>aries_cloudagent.protocols.connections.routes)</code>
148	157
<code>ConnectionResponse</code> (class in <code>ConnectionTarget</code> (class in <code>aries_cloudagent.protocols.connections.messages.connection_response()</code> (in module <code>aries_cloudagent.connections.models.connection_target)</code>),	<code>aries_cloudagent.connections.models.connection_target)</code>
148	129
<code>ConnectionResponse.Meta</code> (class in <code>ConnectionTarget.Meta</code> (class in <code>aries_cloudagent.protocols.connections.messages.connection_response()</code> (in module <code>aries_cloudagent.connections.models.connection_target)</code>),	<code>aries_cloudagent.connections.models.connection_target)</code>
148	129
<code>ConnectionResponseHandler</code> (class in <code>ConnectionTargetSchema</code> (class in <code>aries_cloudagent.protocols.connections.handlers.connection_response_handler()</code> (in module <code>aries_cloudagent.connections.models.connection_target)</code>),	<code>aries_cloudagent.connections.models.connection_target)</code>
145	129
<code>ConnectionResponseSchema</code> (class in <code>ConnectionTargetSchema.Meta</code> (class in <code>aries_cloudagent.protocols.connections.messages.connection_response()</code> (in module <code>aries_cloudagent.connections.models.connection_target)</code>),	<code>aries_cloudagent.connections.models.connection_target)</code>
148	129
<code>ConnectionResponseSchema.Meta</code> (class in <code>content</code> (<code>aries_cloudagent.protocols.basicmessage.messages.basicmessage()</code> (in module <code>aries_cloudagent.protocols.connections.messages.connection_response()</code> (in module <code>aries_cloudagent.connections.models.connection_target)</code>),	<code>aries_cloudagent.connections.models.connection_target)</code>
148	143
<code>connections_accept_invitation()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>CONTEXT (aries_cloudagent.connections.models.diddoc.DIDDoc attribute), 115</code>
157	<code>CONTEXT (aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc attribute), 119</code>
<code>connections_accept_request()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>context (aries_cloudagent.protocols.connections.manager.ConnectionManager attribute), 152</code>
157	<code>context (aries_cloudagent.protocols.credentials.manager.CredentialManager attribute), 167</code>
<code>connections_create_invitation()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>context (aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager attribute), 198</code>
158	<code>context (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager attribute), 219</code>
<code>connections_create_static()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>context (aries_cloudagent.protocols.presentations.manager.PresentationManager attribute), 233</code>
158	<code>context (aries_cloudagent.protocols.routing.manager.RoutingManager attribute), 250</code>
<code>connections_establish_inbound()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>ContextBuilder</code> (class in <code>aries_cloudagent.config.base_context)</code> , 12
158	<code>controller (aries_cloudagent.connections.models.diddoc.PublicKey attribute), 117</code>
<code>connections_receive_invitation()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>controller (aries_cloudagent.connections.models.diddoc.publickey.PublicKey attribute), 121</code>
158	<code>Controller</code> (class in <code>aries_cloudagent.protocols.actionmenu.controller)</code> , 138
<code>connections_remove()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	<code>controllers (aries_cloudagent.core.protocol_registry.ProtocolRegistry attribute), 159</code>
159	
<code>connections_retrieve()</code> (in module <code>aries_cloudagent.protocols.connections.routes)</code>	

attribute), 22
 copy () (aries_cloudagent.config.base.BaseInjector method), 11
 copy () (aries_cloudagent.config.base.BaseSettings method), 11
 copy () (aries_cloudagent.config.injection_context.InjectionContext method), 13
 copy () (aries_cloudagent.config.injector.Injector method), 15
 copy () (aries_cloudagent.config.settings.Settings method), 17
 copy () (aries_cloudagent.messaging.decorators.base.BaseDecorator method), 39
 copy () (aries_cloudagent.messaging.request_context.RequestContext method), 59
 coro_ident () (in module aries_cloudagent.utils.task_queue), 95
 coro_timed () (in module aries_cloudagent.utils.task_queue), 95
 create () (aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator class method), 42
 create () (aries_cloudagent.wallet.indy.IndyWallet method), 106
 create_bound_request () (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager method), 220
 create_credential () (aries_cloudagent.issuer.indy.IndyIssuer method), 25
 create_credential_offer () (aries_cloudagent.issuer.indy.IndyIssuer method), 25
 create_credential_request () (aries_cloudagent.holder.indy.IndyHolder method), 23
 create_did_document () (aries_cloudagent.protocols.connections.manager.ConnectionManager method), 152
 create_exchange_for_proposal () (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager method), 220
 create_exchange_for_request () (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager method), 220
 create_invitation () (aries_cloudagent.protocols.connections.manager.ConnectionManager method), 152
 create_keypair () (in module aries_cloudagent.wallet.crypto), 102
 create_local_did () (aries_cloudagent.wallet.base.BaseWallet method), 96
 create_local_did () (aries_cloudagent.wallet.basic.BasicWallet method), 99
 create_local_did () (aries_cloudagent.wallet.indy.IndyWallet method), 106
 create_offer () (aries_cloudagent.protocols.credentials.manager.CredentialsManager method), 167
 create_offer () (aries_cloudagent.protocols.issue_credential.v1_0.manager.IssueCredentialManager method), 198
 create_outbound () (aries_cloudagent.core.dispatcher.DispatcherResponder method), 21
 create_outbound () (aries_cloudagent.messaging.responder.BaseResponder method), 59
 create_pool_config () (aries_cloudagent.ledger.indy.IndyLedger method), 28
 create_presentation () (aries_cloudagent.holder.indy.IndyHolder method), 23
 create_presentation () (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager method), 220
 create_presentation () (aries_cloudagent.protocols.presentations.manager.PresentationManager method), 233
 create_proposal () (aries_cloudagent.protocols.issue_credential.v1_0.manager.IssueCredentialManager method), 199
 create_public_did () (aries_cloudagent.wallet.base.BaseWallet method), 96
 create_request () (aries_cloudagent.protocols.connections.manager.ConnectionManager method), 153
 create_request () (aries_cloudagent.protocols.credentials.manager.CredentialsManager method), 168
 create_request () (aries_cloudagent.protocols.issue_credential.v1_0.manager.IssueCredentialManager method), 199
 create_request () (aries_cloudagent.protocols.presentations.manager.PresentationManager method), 233
 create_response () (aries_cloudagent.protocols.connections.manager.ConnectionManager method), 153
 create_route_record () (aries_cloudagent.protocols.routing.manager.RoutingManager method), 250
 create_session () (aries_cloudagent.transport.inbound.base.BaseInboundTransport method), 71
 create_session () (aries_cloudagent.transport.inbound.manager.InboundTransportManager method), 74
 create_signing_key () (aries_cloudagent.wallet.base.BaseWallet method), 96
 create_signing_key () (aries_cloudagent.wallet.indy.IndyWallet method), 106

`(aries_cloudagent.wallet.basic.BasicWallet method), 99`
`create_signing_key()`
`(aries_cloudagent.wallet.indy.IndyWallet method), 106`
`create_static_connection()`
`(aries_cloudagent.protocols.connections.manager.ConnectionManager method), 153`
`created (aries_cloudagent.wallet.base.BaseWallet attribute), 97`
`created (aries_cloudagent.wallet.basic.BasicWallet attribute), 99`
`created (aries_cloudagent.wallet.indy.IndyWallet attribute), 106`
`created_at (aries_cloudagent.messaging.models.base_record.BaseRecord attribute), 52`
`created_at (aries_cloudagent.protocols.routing.models.route_record.RouteRecord attribute), 248`
`cred_def_id (aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposalSchema attribute), 193`
`cred_def_id (aries_cloudagent.protocols.present_proof.v1_0.messages.independent_proof_or_elsewise_proof_attr_spec.Schema attribute), 210`
`cred_def_id (aries_cloudagent.protocols.present_proof.v1_0.messages.inner_presentation_preview.PresProofPreviewSchema attribute), 211`
`CredAttrSpec (class in aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_credential_preview), 187`
`CredAttrSpec.Meta (class in aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_credential_preview), 187`
`CredAttrSpecSchema (class in aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_credential_preview), 188`
`CredAttrSpecSchema.Meta (class in aries_cloudagent.protocols.issue_credential.v1_0.messages.inner_credential_preview), 188`
`credential (aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema attribute), 167`
`credential (aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchangeSchema attribute), 198`
`credential_definition_id (aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema attribute), 167`
`credential_definition_id (aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchangeSchema attribute), 198`
`credential_definition_id2schema_id()`
`(aries_cloudagent.ledger.indy.IndyLedger method), 28`
`credential_definitions_created() (in module aries_cloudagent.messaging.credential_definitions.routes), 34`
`credential_definitions_get_credential_definition_id()`
`(in module aries_cloudagent.messaging.credential_definitions.routes), 34`
`credential_definitions_send_credential_definition()`
`(in module aries_cloudagent.messaging.credential_definitions.routes), 35`
`credential_exchange_id (aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema attribute), 166`
`credential_exchange_id (aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchangeSchema attribute), 167`
`credential_exchange_id (aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchangeSchema attribute), 197`
`credential_exchange_id (aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.CredentialExchangeSchema attribute), 198`
`credential_exchange_remove() (in module aries_cloudagent.protocols.issue_credential.v1_0.routes), 204`
`credential_exchange_retrieve() (in module aries_cloudagent.protocols.issue_credential.v1_0.routes), 205`
`credential_exchange_send() (in module aries_cloudagent.protocols.issue_credential.v1_0.routes), 205`
`credential_exchange_send_bound_offer() (in module aries_cloudagent.protocols.issue_credential.v1_0.routes), 205`

205	credential_exchange_send_free_offer()	credential_stored()
	(in module aries_cloudagent.protocols.issue_credential.v1_0.routes), 168	(aries_cloudagent.protocols.credentials.manager.CredentialManager), 168
205	credential_exchange_send_offer()	credential_values
	(in module aries_cloudagent.protocols.credentials.routes), 167	(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema), 167
175	credential_exchange_send_proposal()	CredentialAck
	(in module aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange_proposal), 189	(class in aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange_proposal), 189
206	credential_exchange_send_request()	CredentialAck.Meta
	(in module aries_cloudagent.protocols.credentials.routes), 189	(class in aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange_request), 189
175	credential_exchange_send_request()	CredentialAckHandler
	(in module aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_exchange_request), 189	(class in aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_exchange_request), 189
206	credential_exchange_store()	CredentialAckSchema
	(in module aries_cloudagent.protocols.credentials.routes), 189	(class in aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange_store), 189
175	credential_exchange_store()	CredentialAckSchema.Meta
	(in module aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange_store), 189	(class in aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange_store), 189
206	credential_id()	CredentialDefinitionGetResultsSchema
	(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema), 167	(class in aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_definition_get_results), 167
	credential_id()	CredentialExchange.V10
	(in module aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_exchange), 198	(class in aries_cloudagent.messaging.credential_definitions.routes), 198
	credential_offer()	CredentialExchange.CredentialExchangeSchema
	(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema), 167	(class in aries_cloudagent.messaging.credential_definitions.routes), 167
	credential_offer()	CredentialDefinitionsCreatedResultsSchema
	(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer), 198	(class in aries_cloudagent.messaging.credential_definitions.routes), 198
	credential_preview	CredentialDefinitionSendRequestSchema
	(aries_cloudagent.protocols.credentials.messages.credential_offer), 162	(class in aries_cloudagent.messaging.credential_definitions.routes), 162
	credential_preview	CredentialDefinitionSendResultsSchema
	(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer), 192	(class in aries_cloudagent.messaging.credential_definitions.routes), 192
	credential_proposal	CredentialExchange
	(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal), 193	(class in aries_cloudagent.messaging.credential_definitions.routes), 193
	credential_proposal_dict	CredentialExchange.Meta
	(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal), 198	(class in aries_cloudagent.messaging.credential_definitions.routes), 198
	credential_request	CredentialExchangeListSchema
	(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema), 167	(class in aries_cloudagent.messaging.credential_definitions.routes), 167
	credential_request	CredentialExchangeSchema
	(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request), 198	(class in aries_cloudagent.messaging.credential_definitions.routes), 198
	credential_request_metadata	CredentialExchangeSchema.Meta
	(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema), 167	(class in aries_cloudagent.messaging.credential_definitions.routes), 167
	credential_request_metadata	CredentialIssue
	(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request), 198	(class in aries_cloudagent.messaging.credential_definitions.routes), 198

CredentialIssue	(class in 191	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue_handler	(class in 189
CredentialIssue.Meta	(class in 160	aries_cloudagent.protocols.credentials.messages.credential_issue_handler	(class in 161
CredentialIssue.Meta	(class in 186	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue_handler	(class in 190
CredentialIssueHandler	(class in 170	aries_cloudagent.protocols.credentials.handlers.credential_issue_handler	(class in 159
CredentialIssueHandler	(class in 170	aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler	(class in 186
CredentialIssueRequestSchema	(class in 162	aries_cloudagent.protocols.credentials.routes),	CredentialOfferSchema (class in 169
CredentialIssueResultSchema	(class in 169	aries_cloudagent.protocols.credentials.routes),	CredentialOfferSchema.Meta (class in 191
CredentialIssueSchema	(class in 162	aries_cloudagent.protocols.credentials.messages.credential_issue_handler	CredentialOfferSchema.Meta (class in 161
CredentialIssueSchema	(class in 191	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue_handler	CredentialOfferSchema.Meta (class in 190
CredentialIssueSchema.Meta	(class in 229	aries_cloudagent.protocols.credentials.messages.credential_issue_handler	CredentialOfferSchema.Meta (class in 161
CredentialIssueSchema.Meta	(class in 229	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue_handler	CredentialOfferSchema.Meta (class in 190
CredentialListSchema	(class in 228	aries_cloudagent.protocols.credentials.routes),	CredentialPresentationSchema (class in 170
CredentialManager	(class in 229	aries_cloudagent.protocols.credentials.manager),	CredentialPresentationSchema.Meta (class in 167
CredentialManager	(class in 229	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue_handler	CredentialPreview (class in 198
CredentialManagerError, 168, 200			CredentialPreview.Meta (class in 188
CredentialOffer	(class in 161	aries_cloudagent.protocols.credentials.messages.credential_offer_handler	CredentialPreviewSchema (class in 188
CredentialOffer	(class in 191	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer_handler	CredentialPreviewSchema.Meta (class in 188
CredentialOffer.Meta	(class in 162	aries_cloudagent.protocols.credentials.messages.credential_offer_handler	CredentialPreviewSchema.Meta (class in 189
CredentialOffer.Meta	(class in 162	aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer_handler	CredentialProblemReportRequestSchema (class in 189

171
CredentialProposal (class in credentials_list() (in module
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal),
192
CredentialProposal.Meta (class in credentials_remove() (in module
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal),
193
CredentialProposalHandler (class in CredentialSchema (class in
aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler),
186
CredentialProposalSchema (class in CredentialSendRequestSchema (class in
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal),
193
CredentialProposalSchema.Meta (class in CredentialSendResultSchema (class in
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal),
193
CredentialRequest (class in CredentialStored (class in
aries_cloudagent.protocols.credentials.messages.credential_request),
162
CredentialRequest (class in CredentialStored.Meta (class in
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request),
193
CredentialRequest.Meta (class in CredentialStoredHandler (class in
aries_cloudagent.protocols.credentials.messages.credential_request),
163
CredentialRequest.Meta (class in CredentialStoredSchema (class in
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request),
194
CredentialRequestHandler (class in CredentialStoredSchema.Meta (class in
aries_cloudagent.protocols.credentials.handlers.credential_request_handler),
160
CredentialRequestHandler (class in CredentialStoreRequestSchema (class in
aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler),
187
CredentialRequestResultSchema (class in current_active(aries_cloudagent.utils.task_queue.TaskQueue
aries_cloudagent.protocols.credentials.routes), attribute), 94
171
CredentialRequestSchema (class in current_pending(aries_cloudagent.utils.task_queue.TaskQueue
aries_cloudagent.protocols.credentials.messages.credential_request), attribute), 94
163
CredentialRequestSchema (class in CredentialRequestSchema (class in
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request),
194
CredentialRequestSchema.Meta (class in data(aries_cloudagent.messaging.decorators.attach_decorator.AttachDeco
aries_cloudagent.protocols.credentials.messages.credential_request), attribute), 38
163
CredentialRequestSchema.Meta (class in datetime_to_str() (in module
aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request), aries_cloudagent.messaging.util), 61
194
DebugGroup (class in
aries_cloudagent.config argparse), 9
credentials_attach (aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.CredentialIssueSchema
attribute), 190
credentials_get() (in module decode_pack_message() (in module
aries_cloudagent.protocols.credentials.routes), aries_cloudagent.wallet.crypto), 103

`method`), 81
`ENCODING_MISMATCH`
`(aries_cloudagent.verifier.indy.PreVerifyResult`
`attribute)`, 96
`encrypt_plaintext()` (in module
`aries_cloudagent.wallet.crypto`), 104
`end(aries_cloudagent.protocols.routing.models.paginated.PaginatedSchema`
`attribute)`, 246
`endpoint(aries_cloudagent.connections.models.connection_target.TargetSchema`
`attribute)`, 130
`endpoint(aries_cloudagent.connections.models.diddoc.Service`
`attribute)`, 118
`endpoint(aries_cloudagent.connections.models.diddoc.service.Service`
`attribute)`, 122
`endpoint(aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitationSchema`
`attribute)`, 147
`enqueue()` (`aries_cloudagent.transport.queue.base.BaseMessageQueue`
`method`), 85
`enqueue()` (`aries_cloudagent.transport.queue.basic.BasicMessageQueue`
`method`), 86
`enqueue_message()`
`(aries_cloudagent.transport.outbound.manager.OutboundTransportManager`
`method)`, 81
`enqueue_webhook()`
`(aries_cloudagent.transport.outbound.manager.OutboundTransportManager`
`method)`, 82
`epoch_to_str()` (in module
`aries_cloudagent.messaging.util`), 61
`error_code(aries_cloudagent.core.error.BaseError`
`attribute)`, 21
`error_code(aries_cloudagent.messaging.error.MessageParseError`
`attribute)`, 58
`error_code(aries_cloudagent.messaging.error.MessagePrepareError`
`attribute)`, 58
`error_code(aries_cloudagent.transport.error.MessageEncodeError`
`attribute)`, 86
`error_code(aries_cloudagent.transport.error.MessageParseError`
`attribute)`, 86
`error_msg(aries_cloudagent.connections.models.connection_record.ConnectionRecord`
`attribute)`, 128
`error_msg(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange`
`attribute)`, 167
`error_msg(aries_cloudagent.protocols.issue_credential.v1_0.models.issue_credential_v1_0.CredentialExchange`
`attribute)`, 198
`error_msg(aries_cloudagent.protocols.present_proof.v1_0.models.present_proof_v1_0.CredentialExchange`
`attribute)`, 218
`error_msg(aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange`
`attribute)`, 232
`errmsg(aries_cloudagent.protocols.actionmenu.messages.menu.MenuSchema`
`attribute)`, 133
`escalation_uri(aries_cloudagent.protocols.problem_report.messages.problem_report.ProblemReportSchema`
`attribute)`, 239
`establish_inbound()`
`(aries_cloudagent.protocols.connections.manager.ConnectionManager`
`method)`, 154
`EXAMPLE(aries_cloudagent.messaging.valid.Base64`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.Base64URL`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyCredDefId`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyDID`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyISO8601DateTime`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyPredicate`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyRawPublicKey`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyRevRegId`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndySchemaId`
`attribute)`, 62
`EXAMPLE(aries_cloudagent.messaging.valid.IndyVersion`
`attribute)`, 63
`EXAMPLE(aries_cloudagent.messaging.valid.IntEpoch`
`attribute)`, 63
`EXAMPLE(aries_cloudagent.messaging.valid.SHA256Hash`
`attribute)`, 63
`EXAMPLE(aries_cloudagent.messaging.valid.UUIDFour`
`attribute)`, 63
`execute()` (in module
`aries_cloudagent.commands.help`), 8
`execute()` (in module
`aries_cloudagent.commands.provision`), 8
`execute()` (in module
`aries_cloudagent.commands.start`), 8
`expire_messages()`
`(aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue`
`method)`, 72
`expires_time(aries_cloudagent.messaging.decorators.timing_decorator`
`attribute)`, 46
`exception_record(aries_cloudagent.connections.messages.problem_report.problem_report.message.ProblemReportMessage`
`attribute)`, 239
`exchange(aries_cloudagent.protocols.issue_credential.v1_0.models.issue_credential_v1_0.CredentialExchange`
`method)`, 11
`exchange(aries_cloudagent.protocols.present_proof.v1_0.models.present_proof_v1_0.CredentialExchange`
`method)`, 17
`exchange(aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange`
`method)`, 92
`extract_decorators()`
`(aries_cloudagent.messaging.agent_message.AgentMessageSchema`
`method)`, 154

method), 57
extract_decorators() (aries_cloudagent.messaging.decorators.base.BaseDecoratorSet *attribute*), 39
extract_pack_recipients() (in module aries_cloudagent.wallet.crypto), 104
extract_payload_key() (in module aries_cloudagent.wallet.crypto), 104
F
fetch() (aries_cloudagent.storage.base.BaseStorageRecordSearch *method*), 64
fetch() (aries_cloudagent.storage.basic.BasicStorageRecordSearch *method*), 67
fetch() (aries_cloudagent.storage.indy.IndyStorageRecordSearch *method*), 70
fetch() (in module aries_cloudagent.utils.http), 91
fetch_all() (aries_cloudagent.storage.base.BaseStorageRecordSearch *method*), 65
fetch_connection_targets() (aries_cloudagent.protocols.connections.manager.ConnectionManager *method*), 154
fetch_credential_definition() (aries_cloudagent.ledger.indy.IndyLedger *method*), 28
fetch_did_document() (aries_cloudagent.protocols.connections.manager.ConnectionManager *method*), 154
fetch_genesis_transactions() (in module aries_cloudagent.config.ledger), 15
fetch_schema_by_id() (aries_cloudagent.ledger.indy.IndyLedger *method*), 28
fetch_schema_by_seq_no() (aries_cloudagent.ledger.indy.IndyLedger *method*), 28
fetch_single() (aries_cloudagent.storage.base.BaseStorageRecordSearch *method*), 65
fetch_txn_author_agreement() (aries_cloudagent.ledger.base.BaseLedger *method*), 26
fetch_txn_author_agreement() (aries_cloudagent.ledger.indy.IndyLedger *method*), 28
FetchError, 91
field() (aries_cloudagent.messaging.decorators.base.BaseDecoratorSet *method*), 39
fields (aries_cloudagent.messaging.decorators.base.BaseDecoratorSet *attribute*), 39
file_ext() (in module aries_cloudagent.wallet.plugin), 109
filename (aries_cloudagent.messaging.decorators.attach_decorator_attach_decorator_schema *attribute*), 39
filter (aries_cloudagent.protocols.routing.messages.route_query_request *attribute*), 242
find_connection() (aries_cloudagent.protocols.connections.manager.ConnectionManager *method*), 154
find_did_for_key() (aries_cloudagent.protocols.connections.manager.ConnectionManager *method*), 154
find_inbound_connection() (aries_cloudagent.protocols.connections.manager.ConnectionManager *method*), 154
finished_deliver() (aries_cloudagent.transport.outbound.manager.OutboundTransportManager *method*), 82
finished_encode() (aries_cloudagent.transport.outbound.manager.OutboundTransportManager *method*), 82
fix_hint_txt (aries_cloudagent.protocols.problem_report.message.ProblemReport *attribute*), 239
flush() (aries_cloudagent.cache.base.BaseCache *method*), 6
flush() (aries_cloudagent.cache.basic.BasicCache *method*), 7
flush() (aries_cloudagent.transport.outbound.manager.OutboundTransportManager *method*), 82
flush() (aries_cloudagent.utils.task_queue.TaskQueue *method*), 94
form (aries_cloudagent.protocols.actionmenu.models.menu_option.MenuOption *attribute*), 138
format_did_info() (in module aries_cloudagent.wallet.routes), 111
Forward (class in aries_cloudagent.protocols.routing.messages.forward), 241
Forward.Meta (class in aries_cloudagent.protocols.routing.messages.forward), 241
ForwardHandler (class in aries_cloudagent.protocols.routing.handlers.forward_handler), 240
ForwardInvitation (class in aries_cloudagent.protocols.introduction.messages.forward_invitation), 180
ForwardInvitation.Meta (class in aries_cloudagent.protocols.introduction.messages.forward_invitation), 181
ForwardInvitationHandler (class in aries_cloudagent.protocols.introduction.handlers.forward_invitation_handler), 180
ForwardInvitationSchema (class in aries_cloudagent.protocols.introduction.messages.forward_invitation), 181
ForwardInvitationSchema.Meta (class in

[aries_cloudagent.protocols.introduction.messages.forward_introduction](#), 18
[181](#)
[ForwardSchema](#) (class in [\(aries_cloudagent.wallet.indy.IndyWallet](#)
[aries_cloudagent.protocols.routing.messages.forward\)](#), method), 107
[241](#)
[ForwardSchema.Meta](#) (class in [\(aries_cloudagent.holder.indy.IndyHolder](#)
[aries_cloudagent.protocols.routing.messages.forward\)](#), method), 24
[241](#)
[get_credentials\(\)](#)
[from_indy_dict\(\)](#) ([aries_cloudagent.messaging.decorators.attach_credentials_decorator](#) in [IndyHolder](#)
class method), 36
[from_json\(\)](#) ([aries_cloudagent.connections.models.diddoc.DIDDoc](#) endpoint()) (in module
class method), 116
[aries_cloudagent.ledger.routes](#)), 31
[from_json\(\)](#) ([aries_cloudagent.connections.models.diddoc.DIDDoc](#) endpoint()) (in module
class method), 119
[aries_cloudagent.ledger.routes](#)), 31
[from_json\(\)](#) ([aries_cloudagent.messaging.models.base.BaseModel](#) endpoint_for_did())
class method), 47
[\(aries_cloudagent.ledger.base.BaseLedger](#)
[from_storage\(\)](#) ([aries_cloudagent.messaging.models.base_record.BaseRecord](#)
class method), 50
[get_endpoint_for_did\(\)](#)
[from_url\(\)](#) ([aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation](#) in [IndyLedger](#)
class method), 146
method), 28
[future](#) ([aries_cloudagent.cache.base.CacheKeyLock](#) get_indy_storage()
attribute), 7
[\(aries_cloudagent.ledger.indy.IndyLedger](#)
method), 28
G
[get_int\(\)](#) ([aries_cloudagent.config.base.BaseSettings](#)
method), 12
[GeneralGroup](#) (class in [aries_cloudagent.config.argparse](#)), 9
[get_key_for_did\(\)](#)
[generate_wallet_key\(\)](#) ([aries_cloudagent.wallet.indy.IndyWallet](#)
class method), 107
method), 26
[get\(\)](#) ([aries_cloudagent.cache.base.BaseCache](#) method), 6
[get\(\)](#) ([aries_cloudagent.cache.basic.BasicCache](#) method), 7
[get_active_menu\(\)](#)
[\(aries_cloudagent.protocols.actionmenu.base_service.BaseMenuService](#)
method), 138
[get_active_menu\(\)](#)
[\(aries_cloudagent.protocols.actionmenu.driver_service.DriverMenuService](#)
method), 139
[get_bool\(\)](#) ([aries_cloudagent.config.base.BaseSettings](#) method), 12
[get_cached_key\(\)](#) ([aries_cloudagent.messaging.models.base_record.BaseRecord](#)
class method), 50
[get_connection_targets\(\)](#)
[\(aries_cloudagent.protocols.connections.manager.ConnectionManager](#)
method), 155
[get_credential\(\)](#) ([aries_cloudagent.holder.indy.IndyHolder](#) method), 24
[get_credential_definition\(\)](#)
[\(aries_cloudagent.ledger.indy.IndyLedger](#) method), 97
[get_credential_definition_tag_policy\(\)](#)
[get_credentials\(\)](#)
[get_credentials_for_presentation_request_by_referent\(\)](#)
[get_endpoint_for_did\(\)](#)
[get_int\(\)](#) ([aries_cloudagent.config.base.BaseSettings](#) method), 12
[get_key_for_did\(\)](#)
[get_key_for_did\(\)](#)
[get_latest_txn_author_acceptance\(\)](#) ([aries_cloudagent.ledger.indy.IndyLedger](#)
method), 28
[get_latest_txn_author_acceptance\(\)](#) ([aries_cloudagent.ledger.base.BaseLedger](#)
method), 26
[get_latest_txn_author_acceptance\(\)](#) ([aries_cloudagent.ledger.indy.IndyLedger](#)
method), 29
[get_local_did\(\)](#) ([aries_cloudagent.wallet.base.BaseWallet](#) method), 97
[get_local_did\(\)](#) ([aries_cloudagent.wallet.basic.BasicWallet](#) method), 100
[get_local_did_for_verkey\(\)](#) ([aries_cloudagent.wallet.indy.IndyWallet](#) method), 107
[get_local_did_for_verkey\(\)](#)
[get_local_did_for_verkey\(\)](#)
[get_local_did_for_verkey\(\)](#)
[get_local_dids\(\)](#) ([aries_cloudagent.wallet.base.BaseWallet](#) method), 97

`get_local_dids()` (*aries_cloudagent.wallet.basic.BasicWallet* `method`), 100
`get_local_dids()` (*aries_cloudagent.wallet.indy.IndyWallet* `method`), 107
`get_mime_type()` (*aries_cloudagent.holder.indy.IndyHolder* `method`), 24
`get_one_message_for_key()` (*aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue* `method`), 73
`get_provider()` (*aries_cloudagent.config.injector.Injector* `method`), 15
`get_public_did()` (*aries_cloudagent.wallet.base.BaseWallet* `method`), 97
`get_recipient()` (*aries_cloudagent.protocols.routing.manager.RoutingManager* `method`), 250
`get_record()` (*aries_cloudagent.storage.base.BaseStorage* `method`), 63
`get_record()` (*aries_cloudagent.storage.basic.BasicStorage* `method`), 66
`get_record()` (*aries_cloudagent.storage.indy.IndyStorage* `method`), 68
`get_registered()` (*aries_cloudagent.config.argparse.group* `class method`), 11
`get_registered_transport_for_scheme()` (*aries_cloudagent.transport.outbound.manager.OutboundTransportManager* `method`), 82
`get_routes()` (*aries_cloudagent.protocols.routing.manager.RoutingManager* `method`), 250
`get_running_transport_for_endpoint()` (*aries_cloudagent.transport.outbound.manager.OutboundTransportManager* `method`), 82
`get_running_transport_for_scheme()` (*aries_cloudagent.transport.outbound.manager.OutboundTransportManager* `method`), 82
`get_schema()` (*aries_cloudagent.ledger.indy.IndyLedger* `method`), 29
`get_settings()` (*aries_cloudagent.config.argparse.AdminGroup* `method`), 9
`get_settings()` (*aries_cloudagent.config.argparse.ArgumentGroup* `method`), 9
`get_settings()` (*aries_cloudagent.config.argparse.DebugGroup* `method`), 9
`get_settings()` (*aries_cloudagent.config.argparse.GeneralGroup* `method`), 9
`get_settings()` (*aries_cloudagent.config.argparse.LedgerGroup* `method`), 10
`get_settings()` (*aries_cloudagent.config.argparse.LoggingGroup* `method`), 10
`get_settings()` (*aries_cloudagent.config.argparse.ProtocolGroup* `method`), 10
`get_settings()` (*aries_cloudagent.config.argparse.TransportGroup* `method`), 10
`get_settings()` (*aries_cloudagent.config.argparse.WalletGroup* `method`), 11
`signature()` (*aries_cloudagent.messaging.agent_message.AgentMessage* `method`), 55
`signing_key()` (*aries_cloudagent.wallet.base.BaseWallet* `method`), 97
`signing_key()` (*aries_cloudagent.wallet.basic.BasicWallet* `method`), 100
`signing_key()` (*aries_cloudagent.wallet.indy.IndyWallet* `method`), 107
`stats()` (*aries_cloudagent.core.conductor.Conductor* `method`), 18
`get_tag_map()` (*aries_cloudagent.messaging.models.base_record.BaseRecord* `class method`), 50
`get_transport_instance()` (*aries_cloudagent.transport.inbound.manager.InboundTransportManager* `method`), 75
`get_transport_instance()` (*aries_cloudagent.transport.outbound.manager.OutboundTransportManager* `method`), 82
`get_txn_author_agreement()` (*aries_cloudagent.ledger.base.BaseLedger* `method`), 26
`get_txn_author_agreement()` (*aries_cloudagent.ledger.indy.IndyLedger* `method`), 29
`get_value()` (*aries_cloudagent.config.settings.Settings* `method`), 82
`GetTagPolicyResultSchema` (`class` in *aries_cloudagent.wallet.routes*), 110
`group` (`class` in *aries_cloudagent.config.argparse*), 11
`NAME` (*aries_cloudagent.config.argparse.AdminGroup* `attribute`), 9
`NAME` (*aries_cloudagent.config.argparse.ArgumentGroup* `attribute`), 9
`NAME` (*aries_cloudagent.config.argparse.DebugGroup* `attribute`), 9
`NAME` (*aries_cloudagent.config.argparse.GeneralGroup* `attribute`), 9
`NAME` (*aries_cloudagent.config.argparse.LedgerGroup* `attribute`), 10
`NAME` (*aries_cloudagent.config.argparse.LoggingGroup* `attribute`), 10
`NAME` (*aries_cloudagent.config.argparse.ProtocolGroup* `attribute`), 10
`NAME` (*aries_cloudagent.config.argparse.TransportGroup* `attribute`), 10
`NAME` (*aries_cloudagent.config.argparse.WalletGroup* `attribute`), 10

H

`handle` (`aries_cloudagent.storage.base.BaseStorageRecordSearch` attribute), 65
`handle` (`aries_cloudagent.storage.indy.IndyStorageRecordSearch` attribute), 70
`handle` (`aries_cloudagent.wallet.base.BaseWallet` attribute), 97
`handle` (`aries_cloudagent.wallet.indy.IndyWallet` attribute), 107
`handle` (`aries_cloudagent.messaging.base_handler.BaseHandler` method), 58
`handle` (`aries_cloudagent.protocols.actionmenu.handlers.menu_handler.MenuHandler` method), 131
`handle` (`aries_cloudagent.protocols.actionmenu.handlers.menu_request_handler.MenuRequestHandler` method), 132
`handle` (`aries_cloudagent.protocols.actionmenu.handlers.perform_handler.PerformHandler` method), 132
`handle` (`aries_cloudagent.protocols.basicmessage.handlers.basicmessage_handlers.BasicMessageHandler` method), 142
`handle` (`aries_cloudagent.protocols.connections.handlers.connection_invitation_handler.ConnectionInvitationHandler` method), 144
`handle` (`aries_cloudagent.protocols.connections.handlers.connection_request_handler.ConnectionRequestHandler` method), 144
`handle` (`aries_cloudagent.protocols.connections.handlers.connection_response_handler.ConnectionResponseHandler` method), 145
`handle` (`aries_cloudagent.protocols.credentials.handlers.credential_issue_handler.CredentialIssueHandler` method), 159
`handle` (`aries_cloudagent.protocols.credentials.handlers.credential_offer_handler.CredentialOfferHandler` method), 160
`handle` (`aries_cloudagent.protocols.credentials.handlers.credential_request_handler.CredentialRequestHandler` method), 160
`handle` (`aries_cloudagent.protocols.credentials.handlers.credential_store_handler.CredentialStoreHandler` method), 160
`handle` (`aries_cloudagent.protocols.discovery.handlers.disclose_handler.DiscloseHandler` method), 176
`handle` (`aries_cloudagent.protocols.discovery.handlers.query_handler.QueryHandler` method), 177
`handle` (`aries_cloudagent.protocols.introduction.handlers.forward_invitation_handler.ForwardInvitationHandler` method), 180
`handle` (`aries_cloudagent.protocols.introduction.handlers.invitation_handler.InvitationHandler` method), 180
`handle` (`aries_cloudagent.protocols.introduction.handlers.invitation_request_handler.InvitationRequestHandler` method), 180
`handle` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_ack_handler.CredentialAckHandler` method), 185
`handle` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler.CredentialIssueHandler` method), 186
`handle` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler.CredentialOfferHandler` method), 186
`handle` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler.CredentialProposalHandler` method), 186
`handle` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler.CredentialRequestHandler` method), 187
`handle` (`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler.PresentationAckHandler` method), 207
`handle` (`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_request_handler.PresentationRequestHandler` method), 207
`handle` (`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_revocation_handler.PresentationRevocationHandler` method), 208
`handle` (`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_revocation_request_handler.PresentationRevocationRequestHandler` method), 208
`handle` (`aries_cloudagent.protocols.presentations.handlers.credential_handler.CredentialHandler` method), 228
`handle` (`aries_cloudagent.protocols.presentations.handlers.presentation_handler.PresentationHandler` method), 229
`handle` (`aries_cloudagent.protocols.problem_report.handler.ProblemReportHandler` method), 237
`handle` (`aries_cloudagent.protocols.routing.handlers.forward_handler.ForwardHandler` method), 240
`handle` (`aries_cloudagent.protocols.routing.handlers.route_query_request_handler.RouteQueryRequestHandler` method), 240
`handle` (`aries_cloudagent.protocols.routing.handlers.route_query_response_handler.RouteQueryResponseHandler` method), 240
`handle` (`aries_cloudagent.protocols.routing.handlers.route_update_request_handler.RouteUpdateRequestHandler` method), 240
`handle` (`aries_cloudagent.protocols.routing.handlers.route_update_response_handler.RouteUpdateResponseHandler` method), 241
`handle` (`aries_cloudagent.protocols.trustping.handlers.ping_handler.PingHandler` method), 251
`handle` (`aries_cloudagent.protocols.trustping.handlers.ping_response_handler.PingResponseHandler` method), 252
`handle_message` (`aries_cloudagent.core.dispatcher.Dispatcher` method), 20
`handle_message` (`aries_cloudagent.transport.outbound.base.BaseOutboundHandler` method), 80
`handle_message` (`aries_cloudagent.transport.outbound.http.HttpTransportHandler` method), 81
`handle_message` (`aries_cloudagent.transport.outbound.ws.WsTransportHandler` method), 84
`handle_not_delivered` (`aries_cloudagent.core.conductor.Conductor` method), 19
`handle_not_returned` (`aries_cloudagent.core.conductor.Conductor` method), 19
`handle_str_message` (`aries_cloudagent.protocols.routing.messages.forward.ForwardSearchMessage` attribute), 241
`handler` (`aries_cloudagent.messaging.agent_message.AgentMessage` attribute), 55
`handler_class` (`aries_cloudagent.messaging.agent_message.AgentMessage` attribute), 55
`handler_class` (`aries_cloudagent.protocols.actionmenu.messages.menu_handler.MenuHandler` attribute), 132
`handler_class` (`aries_cloudagent.protocols.actionmenu.messages.menu_request_handler.MenuRequestHandler` attribute), 132
`handler_class` (`aries_cloudagent.protocols.actionmenu.messages.perform_handler.PerformHandler` attribute), 132
`handler_class` (`aries_cloudagent.protocols.basicmessage.handlers.basicmessage_handlers.BasicMessageHandler` attribute), 142
`handler_class` (`aries_cloudagent.protocols.connections.handlers.connection_invitation_handler.ConnectionInvitationHandler` attribute), 144
`handler_class` (`aries_cloudagent.protocols.connections.handlers.connection_request_handler.ConnectionRequestHandler` attribute), 144
`handler_class` (`aries_cloudagent.protocols.connections.handlers.connection_response_handler.ConnectionResponseHandler` attribute), 145
`handler_class` (`aries_cloudagent.protocols.credentials.handlers.credential_issue_handler.CredentialIssueHandler` attribute), 159
`handler_class` (`aries_cloudagent.protocols.credentials.handlers.credential_offer_handler.CredentialOfferHandler` attribute), 160
`handler_class` (`aries_cloudagent.protocols.credentials.handlers.credential_request_handler.CredentialRequestHandler` attribute), 160
`handler_class` (`aries_cloudagent.protocols.credentials.handlers.credential_store_handler.CredentialStoreHandler` attribute), 160
`handler_class` (`aries_cloudagent.protocols.discovery.handlers.disclose_handler.DiscloseHandler` attribute), 176
`handler_class` (`aries_cloudagent.protocols.discovery.handlers.query_handler.QueryHandler` attribute), 177
`handler_class` (`aries_cloudagent.protocols.introduction.handlers.forward_invitation_handler.ForwardInvitationHandler` attribute), 180
`handler_class` (`aries_cloudagent.protocols.introduction.handlers.invitation_handler.InvitationHandler` attribute), 180
`handler_class` (`aries_cloudagent.protocols.introduction.handlers.invitation_request_handler.InvitationRequestHandler` attribute), 180
`handler_class` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_ack_handler.CredentialAckHandler` attribute), 185
`handler_class` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler.CredentialIssueHandler` attribute), 186
`handler_class` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler.CredentialOfferHandler` attribute), 186
`handler_class` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler.CredentialProposalHandler` attribute), 186
`handler_class` (`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler.CredentialRequestHandler` attribute), 187
`handler_class` (`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler.PresentationAckHandler` attribute), 134

[handler_class \(aries_cloudagent.protocols.basicmessage.messages.basicmessage.Message.Meta](#)
[attribute\), 143](#)

[handler_class \(aries_cloudagent.protocols.connection.messages.connect\(AriesIndividualConnectionInvitationMessage.route_up](#)
[attribute\), 146](#)

[handler_class \(aries_cloudagent.protocols.connection.messages.connect\(AriesRequestAgentConnectionRequestMessage.route_up](#)
[attribute\), 148](#)

[handler_class \(aries_cloudagent.protocols.connection.messages.connect\(AriesResponseAgentConnectionResponseMessage.route_up](#)
[attribute\), 148](#)

[handler_class \(aries_cloudagent.protocols.connection.messages.problem_report\(ProblemReportMessage.trustping.messages.ping.Pi](#)
[attribute\), 149](#)

[handler_class \(aries_cloudagent.protocols.credentials.messages.credential_issue\(CredentialIssueMessage.trustping.messages.ping_re](#)
[attribute\), 161](#)

[handler_class \(aries_cloudagent.protocols.credentials.messages.credential_offer\(CredentialOfferMessage](#)
[attribute\), 162](#)

[handler_class \(aries_cloudagent.protocols.credentials.messages.credential_request\(CredentialRequest.Meta](#)
[attribute\), 163](#)

[handler_class \(aries_cloudagent.protocols.credentials.messages.credential_stored\(CredentialStored.Meta](#)
[attribute\), 163](#)

[handler_class \(aries_cloudagent.protocols.discovery.messages.discover\(DiscoverMessage.transport.inbound.delivery_queue.DeliveryQue](#)
[attribute\), 177](#)

[handler_class \(aries_cloudagent.protocols.discovery.messages.enqueue\(MessageQueueAgent.messaging.decorators.attach_decorator.Attach](#)
[attribute\), 178](#)

[handler_class \(aries_cloudagent.protocols.introduction.messages.agent_for_cloud_intentions_for_invitation\(IntentionMessageInboundTransportConfigur](#)
[attribute\), 181](#)

[handler_class \(aries_cloudagent.protocols.introduction.messages.invitation.Invitation.Meta](#)
[attribute\), 182](#)

[handler_class \(aries_cloudagent.protocols.introduction.messages.invitation_request.InvitationRequest.Meta](#)
[attribute\), 183](#)

[handler_class \(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue\(CredentialIssueMessage](#)
[attribute\), 189](#)

[handler_class \(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue\(CredentialIssueMessage](#)
[attribute\), 190](#)

[handler_class \(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer\(CredentialOfferMessage](#)
[attribute\), 191](#)

[handler_class \(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal\(CredentialProposalMessage](#)
[attribute\), 193](#)

[handler_class \(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request\(CredentialRequest.Meta](#)
[attribute\), 194](#)

[handler_class \(aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request\(PresentationRequest.Meta](#)
[attribute\), 213](#)

[handler_class \(aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request\(PresentationRequest.Meta](#)
[attribute\), 214](#)

[handler_class \(aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request\(PresentationRequest.Meta](#)
[attribute\), 215](#)

[handler_class \(aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request\(PresentationRequest.Meta](#)
[attribute\), 215](#)

[handler_class \(aries_cloudagent.protocols.presentation.messages.credential_proposal\(CredentialProposalMessage](#)
[attribute\), 229](#)

[handler_class \(aries_cloudagent.protocols.presentation.messages.presentation_request\(PresentationRequest.Meta](#)
[attribute\), 230](#)

[handler_class \(aries_cloudagent.protocols.problem_report.messages.problem_report\(ProblemReportMessage.transport.inbound.receipt.MessageReceipt](#)
[attribute\), 239](#)

[handler_class \(aries_cloudagent.protocols.routing.messages.forward\(ForwardMessage](#)
[attribute\), 241](#)

`attribute`), 128
`inbound_message_handler()` (`aries_cloudagent.transport.inbound.http.HttpTransport` `method`), 73
`inbound_message_handler()` (`aries_cloudagent.transport.inbound.ws.WsTransport` `method`), 79
`inbound_message_router()` (`aries_cloudagent.core.conductor.Conductor` `method`), 19
`InboundMessage` (class in `aries_cloudagent.transport.inbound.message`), 75
`InboundSession` (class in `aries_cloudagent.transport.inbound.session`), 77
`InboundTransportConfiguration` (class in `aries_cloudagent.transport.inbound.base`), 71
`InboundTransportError`, 72
`InboundTransportManager` (class in `aries_cloudagent.transport.inbound.manager`), 74
`InboundTransportRegistrationError`, 72
`InboundTransportSetupError`, 72
`INCOMPLETE` (`aries_cloudagent.verifier.indy.PreVerifyResult` `attribute`), 96
`indy_cred_req()` (`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request.CredentialRequest` `method`), 194
`indy_credential()` (`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue.CredentialIssue` `method`), 190
`indy_dict` (`aries_cloudagent.messaging.decorators.attach_decorator.attach_decorator` `attribute`), 37
`indy_offer()` (`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer.CredentialOffer` `method`), 191
`indy_proof()` (`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation.Presentation` `method`), 213
`indy_proof_req_preview2indy_requested_creds()` (`aries_cloudagent.storage.indy`), 69
`indy_proof_request()` (`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequest` `method`), 216
`IndyCredDefId` (class in `aries_cloudagent.messaging.valid`), 62
`IndyDID` (class in `aries_cloudagent.messaging.valid`), 62
`IndyErrorHandler` (class in `aries_cloudagent.ledger.indy`), 27
`IndyHolder` (class in `aries_cloudagent.holder.indy`), 23
`IndyISO8601DateTime` (class in `aries_cloudagent.messaging.valid`), 62
`IndyIssuer` (class in `aries_cloudagent.issuer.indy`), 25
`IndyLedger` (class in `aries_cloudagent.ledger.indy`), 27
`IndyPredicate` (class in `aries_cloudagent.messaging.valid`), 62
`IndyProofReqAttrSpecSchema` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 222
`IndyProofReqNonRevoked` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 222
`IndyProofReqPredSpecSchema` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 222
`IndyProofReqSpecRestrictionsSchema` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 223
`IndyProofRequestSchema` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 223
`IndyRawPublicKey` (class in `aries_cloudagent.messaging.valid`), 62
`IndyRequestedCredsRequestedAttrSchema` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 224
`IndyRequestedCredsRequestedPredSchema` (class in `aries_cloudagent.protocols.present_proof.v1_0.routes`), 224
`IndyRevRegId` (class in `aries_cloudagent.messaging.valid`), 62
`IndySchemaId` (class in `aries_cloudagent.messaging.valid`), 62
`IndyStorage` (class in `aries_cloudagent.storage.indy`), 69
`IndyStorageRecordSearch` (class in `aries_cloudagent.storage.indy`), 69
`IndyVersion` (class in `aries_cloudagent.messaging.valid`), 62
`IndyWallet` (class in `aries_cloudagent.wallet.indy`), 105
`init_argument_parser()` (in module `aries_cloudagent.commands.start`), 8
`init_context()` (`aries_cloudagent.core.plugin_registry.PluginRegistry` `method`), 22
`initiator` (`aries_cloudagent.connections.models.connection_record.ConnectionRecord` `attribute`), 128
`initiator` (`aries_cloudagent.protocols.credentials.models.credential_exchange` `attribute`), 167

[InvitationRequestSchema.Meta](#) (class in [aries_cloudagent.protocols.introduction.messages.invitation](#)), 106
[InvitationRequestSchema](#) (class in [aries_cloudagent.protocols.introduction.messages.invitation](#)), 183
[InvitationResultSchema](#) (class in [aries_cloudagent.protocols.connections.routes](#)), 157
[InvitationSchema](#) (class in [aries_cloudagent.protocols.introduction.messages.invitation](#)), 182
[InvitationSchema.Meta](#) (class in [aries_cloudagent.protocols.introduction.messages.invitation](#)), 182
[invite_message_handler\(\)](#) ([aries_cloudagent.transport.inbound.http.HttpTransport](#) method), 73
[is_multiuse_invitation](#) ([aries_cloudagent.connections.models.connection_record.ConnectionRecord](#) attribute), 127
[is_ready](#) ([aries_cloudagent.connections.models.connection_record.ConnectionRecord](#) attribute), 127
[issue](#) ([aries_cloudagent.protocols.credentials.messages.credential_issue.CredentialIssueSchema](#) attribute), 161
[issue_credential\(\)](#) ([aries_cloudagent.protocols.credentials.manager.CredentialManager](#) method), 168
[issue_credential\(\)](#) ([aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager](#) method), 199
[issuer_did](#) ([aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposalSchema](#) attribute), 193
[IssuerError](#), 25
J
[join\(\)](#) ([aries_cloudagent.transport.queue.base.BaseMessageQueue](#) method), 85
[join\(\)](#) ([aries_cloudagent.transport.queue.basic.BasicMessageQueue](#) method), 86
[json](#) ([aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorData](#) attribute), 37
[json_](#) ([aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorDataSchema](#) attribute), 38
[JsonWireFormat](#) (class in [aries_cloudagent.transport.wire_format](#)), 89
K
[KEY_DERIVATION_ARGON2I_INT](#) ([aries_cloudagent.wallet.indy.IndyWallet](#) attribute), 106
[KEY_DERIVATION_ARGON2I_MOD](#) ([aries_cloudagent.wallet.indy.IndyWallet](#) attribute), 106
[KEY_DERIVATION_RAW](#) ([aries_cloudagent.wallet.indy.IndyWallet](#) attribute), 106

[load_command\(\)](#) (in module [aries_cloudagent.config.argparse](#)), 10
[aries_cloudagent.commands](#)), 7
[load_decorator\(\)](#) ([aries_cloudagent.messaging.decorators.base.BaseDecoratorSet](#) method), 39
[load_message_types\(\)](#) ([aries_cloudagent.admin.server.AdminServer](#) method), 5
[load_module\(\)](#) ([aries_cloudagent.core.plugin_registry.PluginRegistry](#) method), 22
[load_plugins\(\)](#) ([aries_cloudagent.config.default_context.DefaultContextBuilder](#) method), 13
[load_postgres_plugin\(\)](#) (in module [aries_cloudagent.wallet.plugin](#)), 109
[load_resource\(\)](#) (in module [aries_cloudagent.config.logging](#)), 16
[load_subclass_of\(\)](#) ([aries_cloudagent.utils.classloader.ClassLoader](#) class method), 90
[locale](#) ([aries_cloudagent.messaging.decorators.localization_decorator.LocalizationDecoratorSchema](#) attribute), 41
[locale](#) ([aries_cloudagent.protocols.problem_report.message.ProblemReportSchema](#) attribute), 239
[localizable](#) ([aries_cloudagent.messaging.decorators.localization_decorator.LocalizationDecoratorSchema](#) attribute), 41
[LocalizationDecorator](#) (class in [aries_cloudagent.messaging.decorators.localization_decorator](#)), 40
[LocalizationDecorator.Meta](#) (class in [aries_cloudagent.messaging.decorators.localization_decorator](#)), 40
[LocalizationDecoratorSchema](#) (class in [aries_cloudagent.messaging.decorators.localization_decorator](#)), 40
[LocalizationDecoratorSchema.Meta](#) (class in [aries_cloudagent.messaging.decorators.localization_decorator](#)), 41
[log\(\)](#) ([aries_cloudagent.utils.stats.Collector](#) method), 92
[log\(\)](#) ([aries_cloudagent.utils.stats.Stats](#) method), 92
[log_state\(\)](#) ([aries_cloudagent.messaging.models.base_record.BaseRecord](#) class method), 50
[LOG_STATE_FLAG](#) ([aries_cloudagent.connections.models.connection_record.ConnectionRecord](#) attribute), 126
[LOG_STATE_FLAG](#) ([aries_cloudagent.messaging.models.base_record.BaseRecord](#) attribute), 49
[LOG_STATE_FLAG](#) ([aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange](#) attribute), 166
[LOG_STATE_FLAG](#) ([aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange](#) attribute), 232
[log_task\(\)](#) ([aries_cloudagent.core.dispatcher.Dispatcher](#) method), 20
[LoggingConfigurator](#) (class in [aries_cloudagent.config.logging](#)), 15
[LoggingGroup](#) (class in [aries_cloudagent.config.logging](#)), 15

```

MenuFormSchema.Meta          (class          in          method), 73
    aries_cloudagent.protocols.actionmenu.models.menu_form_id (aries_cloudagent.messaging.decorators.please_ack_decorator
    135                                                         attribute), 41
MenuHandler                  (class          in message_receipt (aries_cloudagent.messaging.request_context.RequestContext),
    aries_cloudagent.protocols.actionmenu.handlers.menu_handler attribute), 59
    131
MenuJsonSchema               (class          in          attribute), 55
    aries_cloudagent.protocols.actionmenu.routes), message_type (aries_cloudagent.protocols.actionmenu.messages.menu_message)
    139                                                         attribute), 133
MenuOption                   (class          in message_type (aries_cloudagent.protocols.actionmenu.messages.menu_message)
    aries_cloudagent.protocols.actionmenu.models.menu_option attribute), 133
    137                                                         message_type (aries_cloudagent.protocols.actionmenu.messages.perform_message)
MenuOption.Meta              (class          in          attribute), 134
    aries_cloudagent.protocols.actionmenu.models.menu_option message_type (aries_cloudagent.protocols.basicmessage.messages.basic_message)
    137                                                         attribute), 143
MenuOptionSchema             (class          in message_type (aries_cloudagent.protocols.connections.messages.connection_message)
    aries_cloudagent.protocols.actionmenu.models.menu_option attribute), 146
    137                                                         message_type (aries_cloudagent.protocols.connections.messages.connection_message)
MenuOptionSchema.Meta        (class          in          attribute), 148
    aries_cloudagent.protocols.actionmenu.models.menu_option message_type (aries_cloudagent.protocols.connections.messages.connection_message)
    137                                                         attribute), 148
MenuRequest                  (class          in message_type (aries_cloudagent.protocols.connections.messages.problem_message)
    aries_cloudagent.protocols.actionmenu.messages.menu_request attribute), 149
    133                                                         message_type (aries_cloudagent.protocols.credentials.messages.credentials_message)
MenuRequest.Meta             (class          in          attribute), 161
    aries_cloudagent.protocols.actionmenu.messages.menu_request message_type (aries_cloudagent.protocols.credentials.messages.credentials_message)
    133                                                         attribute), 162
MenuRequestHandler           (class          in message_type (aries_cloudagent.protocols.credentials.messages.credentials_message)
    aries_cloudagent.protocols.actionmenu.handlers.menu_request attribute), 163
    131                                                         message_type (aries_cloudagent.protocols.credentials.messages.credentials_message)
MenuRequestSchema            (class          in          attribute), 163
    aries_cloudagent.protocols.actionmenu.messages.menu_request message_type (aries_cloudagent.protocols.discovery.messages.discovery_message)
    133                                                         attribute), 177
MenuRequestSchema.Meta       (class          in message_type (aries_cloudagent.protocols.discovery.messages.query_message)
    aries_cloudagent.protocols.actionmenu.messages.menu_request attribute), 178
    133                                                         message_type (aries_cloudagent.protocols.introduction.messages.forward_message)
MenuSchema                   (class          in          attribute), 181
    aries_cloudagent.protocols.actionmenu.messages.menu_message message_type (aries_cloudagent.protocols.introduction.messages.invitation_message)
    133                                                         attribute), 182
MenuSchema.Meta              (class          in message_type (aries_cloudagent.protocols.introduction.messages.invitation_message)
    aries_cloudagent.protocols.actionmenu.messages.menu), attribute), 183
    133                                                         message_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0_message)
message                      (aries_cloudagent.core.error.BaseError attribute), 189
    attribute), 21                                                         message_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0_message)
message (aries_cloudagent.messaging.request_context.RequestContext attribute), 190
    attribute), 59                                                         message_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0_message)
message (aries_cloudagent.protocols.introduction.messages.forward_message) ForwardInvitationSchema
    attribute), 181                                                         message_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0_message)
message (aries_cloudagent.protocols.introduction.messages.invitation_message) InvitationSchema
    attribute), 182                                                         message_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0_message)
message (aries_cloudagent.protocols.introduction.messages.invitation_message) InvitationRequestSchema
    attribute), 183                                                         message_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0_message)
message_count_for_key ()
    attribute), 188
    (aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue) (aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof_v1_0_message)

```


attribute), 211		model_class (aries_cloudagent.connections.models.connection_record,
message_type (aries_cloudagent.protocols.present_proof.v1_0.messages.presentation.Presentation.Meta		attribute), 128
attribute), 213		model_class (aries_cloudagent.connections.models.connection_target.C
message_type (aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack.PresentationAck.Meta		attribute), 129
attribute), 214		model_class (aries_cloudagent.messaging.ack.message.AckSchema.Me
message_type (aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal.PresentationProposal.Meta		attribute), 130
attribute), 215		model_class (aries_cloudagent.messaging.agent_message.AgentMessag
message_type (aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request.PresentationRequest.Meta		attribute), 131
attribute), 215		model_class (aries_cloudagent.messaging.decorators.attach_decorator.
message_type (aries_cloudagent.protocols.presentations.messages.credential_presentation.CredentialPresentation.Meta		attribute), 169
attribute), 229		model_class (aries_cloudagent.messaging.decorators.attach_decorator.
message_type (aries_cloudagent.protocols.presentations.messages.presentation_request.PresentationRequest.Meta		attribute), 170
attribute), 230		model_class (aries_cloudagent.messaging.decorators.localization_deco
message_type (aries_cloudagent.protocols.problem_report.messages.problem_report.ProblemReport.Meta		attribute), 239
attribute), 239		model_class (aries_cloudagent.messaging.decorators.please_ack_decor
message_type (aries_cloudagent.protocols.routing.messages.forward_and_invoke.ForwardAndInvokeMeta		attribute), 241
attribute), 241		model_class (aries_cloudagent.messaging.decorators.signature_decora
message_type (aries_cloudagent.protocols.routing.messages.route_query.RouteQueryRequest.Meta		attribute), 242
attribute), 242		model_class (aries_cloudagent.messaging.decorators.thread_decorator
message_type (aries_cloudagent.protocols.routing.messages.route_query_response.RouteQueryResponse.Meta		attribute), 243
attribute), 243		model_class (aries_cloudagent.messaging.decorators.timing_decorator
message_type (aries_cloudagent.protocols.routing.messages.route_update_request.RouteUpdateRequest.Meta		attribute), 244
attribute), 244		model_class (aries_cloudagent.messaging.decorators.transport_decora
message_type (aries_cloudagent.protocols.routing.messages.route_update_response.RouteUpdateResponse.Meta		attribute), 244
attribute), 244		model_class (aries_cloudagent.messaging.models.base.BaseModelSche
message_type (aries_cloudagent.protocols.trustping.messages.ping.Ping.Meta		attribute), 252
attribute), 252		model_class (aries_cloudagent.protocols.actionmenu.messages.menu.M
message_type (aries_cloudagent.protocols.trustping.messages.ping_response.PingResponse.Meta		attribute), 253
attribute), 253		model_class (aries_cloudagent.protocols.actionmenu.messages.menu_r
message_types (aries_cloudagent.core.protocol_registry.ProtocolRegistry), 133		attribute), 133
attribute), 22		model_class (aries_cloudagent.protocols.actionmenu.messages.perform
MessageEncodeError, 86		attribute), 134
MessageParseError, 58, 86		model_class (aries_cloudagent.protocols.actionmenu.models.menu_for
MessagePrepareError, 58		attribute), 135
MessageReceipt	(class in	model_class (aries_cloudagent.protocols.actionmenu.models.menu_for
aries_cloudagent.transport.inbound.receipt),		attribute), 136
76		model_class (aries_cloudagent.protocols.actionmenu.models.menu_opt
metadata (aries_cloudagent.wallet.base.DIDInfo at-		tribute), 137
tribute), 99		model_class (aries_cloudagent.protocols.basicmessage.messages.basica
metadata (aries_cloudagent.wallet.base.KeyInfo at-		tribute), 143
tribute), 99		model_class (aries_cloudagent.protocols.connections.messages.conne
mime_type (aries_cloudagent.messaging.decorators.attach_decorator.AttachDecoratorSchema		attribute), 147
attribute), 39		model_class (aries_cloudagent.protocols.connections.messages.conne
mime_type (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_preview.CredAttrSpecSchema		attribute), 148
attribute), 188		model_class (aries_cloudagent.protocols.connections.messages.conne
mime_type (aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_preview.PresAttrSpecSchema		attribute), 150
attribute), 210		model_class (aries_cloudagent.protocols.connections.messages.proble
mime_types () (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_preview.CredentialPreview		attribute), 150
method), 188		model_class (aries_cloudagent.protocols.connections.models.connectio
MockResponder	(class in	tribute), 150
aries_cloudagent.messaging.responder),		model_class (aries_cloudagent.protocols.credentials.messages.credenti
60		tribute), 161
Model (aries_cloudagent.messaging.models.base.BaseModelSchema		class (aries_cloudagent.protocols.credentials.messages.credenti
tribute), 48		tribute), 162

name (aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof.v1_0.messages.BaseStorageRecordSearch attribute), 211
name (aries_cloudagent.wallet.base.BaseWallet attribute), 97
name (aries_cloudagent.wallet.basic.BasicWallet attribute), 100
name (aries_cloudagent.wallet.indy.IndyWallet attribute), 107
next() (aries_cloudagent.utils.repeat.RepeatAttempt method), 91
next_interval (aries_cloudagent.utils.repeat.RepeatAttempt attribute), 91
next_interval() (aries_cloudagent.utils.repeat.RepeatSequence method), 92
now() (aries_cloudagent.utils.stats.Timer class options (aries_cloudagent.protocols.actionmenu.messages.menu.MenuSchema attribute), 133
nym_to_did() (aries_cloudagent.ledger.base.BaseLedger options (aries_cloudagent.storage.base.BaseStorageRecordSearch attribute), 65
nym_to_did() (aries_cloudagent.ledger.indy.IndyLedger options (aries_cloudagent.ledger.routes.AMLRecordSchema attribute), 30
options (aries_cloudagent.ledger.routes.TAAAcceptanceSchema attribute), 30
offer_json (aries_cloudagent.protocols.credentials.messages.credentials.offer.CredentialOfferSchema attribute), 162
offers_attach (aries_cloudagent.protocols.issue_credential.v1.messages.issue_credential.v1.messages.CredentialOfferSchema attribute), 192
offset (aries_cloudagent.protocols.routing.models.pagination.PaginationSchema attribute), 245
OK (aries_cloudagent.verifier.indy.PreVerifyResult options (aries_cloudagent.ledger.routes.TAAInfoSchema attribute), 31
ok_did() (in module options (aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSchema attribute), 33
aries_cloudagent.connections.models.diddoc.util), options (aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSchema attribute), 123
older_than() (aries_cloudagent.transport.inbound.delivery_queue.QueueMessage options (aries_cloudagent.messaging.credential_definitions.routes.CredentialDefinitionSchema attribute), 73
on (aries_cloudagent.messaging.decorators.please_ack_decorator.PleaseAckDecoratorSchema attribute), 41
on_startup() (aries_cloudagent.admin.server.AdminServer attribute), 5
open() (aries_cloudagent.ledger.indy.IndyLedger attribute), 34
method), 29 options (aries_cloudagent.messaging.schemas.routes.SchemaGetResultsSchema attribute), 53
open() (aries_cloudagent.storage.base.BaseStorageRecordSearch attribute), 65 options (aries_cloudagent.messaging.schemas.routes.SchemaSchema attribute), 53
open() (aries_cloudagent.storage.basic.BasicStorageRecordSearch attribute), 67 options (aries_cloudagent.messaging.schemas.routes.SchemasCreatedResultsSchema attribute), 54
open() (aries_cloudagent.storage.indy.IndyStorageRecordSearch attribute), 70 options (aries_cloudagent.messaging.schemas.routes.SchemaSendRequestSchema attribute), 53
open() (aries_cloudagent.wallet.base.BaseWallet attribute), 97 options (aries_cloudagent.messaging.schemas.routes.SchemaSendResultsSchema attribute), 54
open() (aries_cloudagent.wallet.basic.BasicWallet attribute), 100 options (aries_cloudagent.protocols.actionmenu.routes.MenuJsonSchema attribute), 140
open() (aries_cloudagent.wallet.indy.IndyWallet attribute), 107 options (aries_cloudagent.protocols.actionmenu.routes.PerformRequestSchema attribute), 140

[\(aries_cloudagent.protocols.actionmenu.driver_service.DriverMenuService method\), 139](#)
[PerformHandler \(class in aries_cloudagent.messaging.decorators.please_ack_decorator\), aries_cloudagent.protocols.actionmenu.handlers.perform_handler\), 132](#)
[PerformRequestSchema \(class in aries_cloudagent.messaging.decorators.please_ack_decorator\), aries_cloudagent.protocols.actionmenu.routes\), 41](#)
[PerformSchema \(class in attribute\), 22](#)
[aries_cloudagent.protocols.actionmenu.messages.perform\), 134](#)
[PerformRegistry \(class in aries_cloudagent.core.plugin_registry\), 22](#)
[PerformSchema.Meta \(class in plugins \(aries_cloudagent.core.plugin_registry.PluginRegistry aries_cloudagent.protocols.actionmenu.messages.perform\), attribute\), 22](#)
[134](#)
[plugins_handler\(\)](#)
[Ping \(class in aries_cloudagent.protocols.trustping.messages.ping\), \(aries_cloudagent.admin.server.AdminServer method\), 5](#)
[252](#)
[Ping.Meta \(class in populate_decorators\(\) aries_cloudagent.protocols.trustping.messages.ping\), \(aries_cloudagent.messaging.agent_message.AgentMessageSchema method\), 57](#)
[252](#)
[PingHandler \(class in port \(aries_cloudagent.transport.inbound.base.InboundTransportConfiguration attribute\), 72](#)
[aries_cloudagent.protocols.trustping.handlers.ping_handler\), 251](#)
[PingRequestResponseSchema \(class in method\), 127](#)
[aries_cloudagent.protocols.trustping.routes\), post_save\(\) \(aries_cloudagent.connections.models.connection_record.BaseRecord method\), 50](#)
[254](#)
[PingRequestSchema \(class in post_save\(\) \(aries_cloudagent.protocols.credentials.models.credential.BaseRecord method\), 166](#)
[aries_cloudagent.protocols.trustping.routes\), posture \(aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_request\), 210](#)
[253](#)
[PingResponse \(class in attribute\), 210](#)
[aries_cloudagent.protocols.trustping.messages.ping_response\), 253](#)
[verify\(\) \(aries_cloudagent.verifier.indy.IndyVerifier static method\), 95](#)
[PingResponse.Meta \(class in predicate \(aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_request\), attribute\), 211](#)
[253](#)
[predicates \(aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_request\), 212](#)
[PingResponseHandler \(class in attribute\), 212](#)
[aries_cloudagent.protocols.trustping.handlers.ping_response_handler\), 252](#)
[aries_cloudagent.messaging.decorators.base.BaseDecoratorSet attribute\), 39](#)
[PingResponseSchema \(class in prefix_tag_filter\(\) aries_cloudagent.messaging.models.base_record.BaseRecord class method\), 50](#)
[253](#)
[PingResponseSchema.Meta \(class in prepare_disclosed\(\) aries_cloudagent.core.protocol_registry.ProtocolRegistry method\), 22](#)
[253](#)
[PingSchema \(class in prepare_pack_recipient_keys\(\) \(in module aries_cloudagent.wallet.crypto\), 104](#)
[252](#)
[aries_cloudagent.protocols.trustping.messages.ping\), prepare_send\(\) \(aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential\), 200](#)
[PingSchema.Meta \(class in method\), 200](#)
[253](#)
[aries_cloudagent.protocols.trustping.messages.ping\), PresAttrSpec \(class in aries_cloudagent.protocols.present_proof.v1_0.messages.inner.presentation_request\), 209](#)
[PleaseAckDecorator \(class in attribute\), 41](#)
[aries_cloudagent.messaging.decorators.please_ack_decorator\), 41](#)
[PleaseAckDecorator.Meta \(class in attribute\), 209](#)
[aries_cloudagent.messaging.decorators.please_ack_decorator\), 209](#)
[PleaseAckDecorator.Meta \(class in attribute\), 209](#)
[aries_cloudagent.messaging.decorators.please_ack_decorator\), 209](#)
[PleaseAckDecorator.Posture \(class in attribute\), 209](#)

`aries_cloudagent.protocols.present_proof.v1_0.messages.incoming.presentation_routes`,
209 236
`PresAttrSpecSchema` (class in `presentation_exchange_retrieve()` (in module `aries_cloudagent.protocols.present_proof.v1_0.messages.incoming.presentation_routes`),
210 226
`PresAttrSpecSchema.Meta` (class in `presentation_exchange_retrieve()` (in module `aries_cloudagent.protocols.present_proof.v1_0.messages.incoming.presentation_routes`),
210 236
`presentation` (`aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange_v1_0.PresentationExchangeSchema`
attribute), 219 (in module `aries_cloudagent.protocols.present_proof.v1_0.routes`)
`presentation` (`aries_cloudagent.protocols.presentations.message226.CredentialPresentationSchema`
attribute), 230 `presentation_exchange_send_credential_presentation`
`presentation` (`aries_cloudagent.protocols.presentations.models.presentation_exchange_v1_0.PresentationExchangeSchema`,
attribute), 232 236
`Presentation` (class in `presentation_exchange_send_free_request()`
`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_routes`),
212 227
`Presentation.Meta` (class in `presentation_exchange_send_presentation()`
`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_routes`),
213 227
`presentation_exchange_create_request()` `presentation_exchange_send_proposal()`
(in module `aries_cloudagent.protocols.present_proof.v1_0.routes`), (in module `aries_cloudagent.protocols.present_proof.v1_0.routes`)
225 227
`presentation_exchange_create_request()` `presentation_exchange_send_request()` (in
(in module `aries_cloudagent.protocols.presentations.routes`), (in module `aries_cloudagent.protocols.presentations.routes`),
235 236
`presentation_exchange_credentials_list()` `presentation_exchange_verify_credential_presentation`
(in module `aries_cloudagent.protocols.present_proof.v1_0.routes`), (in module `aries_cloudagent.protocols.presentations.routes`),
226 236
`presentation_exchange_credentials_list()` `presentation_exchange_verify_presentation()`
(in module `aries_cloudagent.protocols.presentations.routes`), (in module `aries_cloudagent.protocols.present_proof.v1_0.routes`)
235 227
`presentation_exchange_id` `presentation_proposal`
(`aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange_v1_0.PresentationExchangeSchema`,
attribute), 218 (attribute), 215
`presentation_exchange_id` `presentation_proposal_dict`
(`aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange_v1_0.PresentationExchangeSchema`,
attribute), 219 (attribute), 219
`presentation_exchange_id` `presentation_request`
(`aries_cloudagent.protocols.presentations.models.presentation_exchange_v1_0.PresentationExchangeSchema`,
attribute), 232 (attribute), 219
`presentation_exchange_id` `presentation_request`
(`aries_cloudagent.protocols.presentations.models.presentation_exchange_v1_0.PresentationExchangeSchema`,
attribute), 232 (attribute), 232
`presentation_exchange_list()` (in module `PresentationAck` (class in
`aries_cloudagent.protocols.present_proof.v1_0.routes`), `aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_routes`),
226 213
`presentation_exchange_list()` (in module `PresentationAck.Meta` (class in
`aries_cloudagent.protocols.presentations.routes`), `aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_routes`),
235 214
`presentation_exchange_remove()` (in module `PresentationAckHandler` (class in
`aries_cloudagent.protocols.present_proof.v1_0.routes`), `aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_routes`),
226 207
`presentation_exchange_remove()` (in module `PresentationAckSchema` (class in

PresPredSpec.Meta (class in process_queued() (aries_cloudagent.transport.outbound.manager.OutboundTransport),
 aries_cloudagent.protocols.present_proof.v1_0.messages.innervation_preview),
 210 process_undelivered()

PresPredSpecSchema (class in (aries_cloudagent.transport.inbound.manager.InboundTransport),
 aries_cloudagent.protocols.present_proof.v1_0.messages.innervation_preview),
 210 ProtocolDescriptorSchema (class in

PresPredSpecSchema.Meta (class in aries_cloudagent.protocols.discovery.messages.disclose),
 aries_cloudagent.protocols.present_proof.v1_0.messages.innervation_preview),
 210 ProtocolGroup (class in

PreVerifyResult (class in aries_cloudagent.config.argparse), 10
 aries_cloudagent.verifier.indy), 96 ProtocolRegistry (class in

print_banner() (aries_cloudagent.config.logging.LoggingConfiguration.aries_cloudagent.core.protocol_registry),
 class method), 16 22

priority (aries_cloudagent.connections.models.diddoc.ServiceProtocols (aries_cloudagent.core.protocol_registry.ProtocolRegistry
 attribute), 118 attribute), 22

priority (aries_cloudagent.connections.models.diddoc.ServiceProtocols (aries_cloudagent.protocols.discovery.messages.disclose.Disclose
 attribute), 122 attribute), 177

problem_code (aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportSchema
 attribute), 150 (aries_cloudagent.core.protocol_registry.ProtocolRegistry

problem_items (aries_cloudagent.protocols.problem_report.messages.problem_report.ProblemReportSchema
 attribute), 239 attribute), 122

ProblemReport (class in method), 11
 aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportSchema (aries_cloudagent.config.provider.CachedProvider
 149 method), 16

ProblemReport (class in provide() (aries_cloudagent.config.provider.ClassProvider
 aries_cloudagent.protocols.problem_report.message), method), 17
 237 provide() (aries_cloudagent.config.provider.InstanceProvider

ProblemReport.Meta (class in method), 17
 aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportSchema (aries_cloudagent.config.provider.StatsProvider
 149 method), 17

ProblemReport.Meta (class in provide() (aries_cloudagent.ledger.provider.LedgerProvider
 aries_cloudagent.protocols.problem_report.message), method), 30
 239 provide() (aries_cloudagent.storage.provider.StorageProvider

ProblemReportHandler (class in method), 70
 aries_cloudagent.protocols.problem_report.handler), provide() (aries_cloudagent.wallet.provider.WalletProvider
 237 method), 110

ProblemReportReason (class in ProviderError, 12
 aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportReason (in module
 149 aries_cloudagent.commands.provision), 8

ProblemReportSchema (class in ProvisionError, 8
 aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportSchema (aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator
 149 attribute), 44

ProblemReportSchema (class in pthid(aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator
 aries_cloudagent.protocols.problem_report.message), attribute), 44
 239 pubkey (aries_cloudagent.connections.models.diddoc.DIDDoc

ProblemReportSchema.Meta (class in attribute), 116
 aries_cloudagent.protocols.connections.messages.problem_report.ProblemReportSchema (aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc
 149 attribute), 119

ProblemReportSchema.Meta (class in PublicKey (class in
 aries_cloudagent.protocols.problem_report.message), aries_cloudagent.connections.models.diddoc),
 239 117

process_inbound() PublicKey (class in
 (aries_cloudagent.transport.inbound.session.InboundSession.aries_cloudagent.connections.models.diddoc.publickey),
 method), 79 120

[PublicKeyType](#) (class in [raw_credential](#) ([aries_cloudagent.protocols.credentials.models.credentials.models](#)), [aries_cloudagent.connections.models.diddoc](#)), [attribute](#), 167
[117](#)
[PublicKeyType](#) (class in [raw_credential](#) ([aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager](#)), [aries_cloudagent.connections.models.diddoc.publickey](#)), [attribute](#), 198
[121](#)
[put\(\)](#) ([aries_cloudagent.utils.task_queue.TaskQueue](#) [method](#)), 94
[put_task\(\)](#) ([aries_cloudagent.core.dispatcher.Dispatcher](#) [method](#)), 20
[ready](#) ([aries_cloudagent.utils.task_queue.TaskQueue](#) [attribute](#)), 94
[receive\(\)](#) ([aries_cloudagent.transport.inbound.session.InboundSession](#) [method](#)), 79
[query](#) ([aries_cloudagent.protocols.discovery.messages.query.QuerySchema](#) [attribute](#)), 178
[Query](#) (class in [aries_cloudagent.protocols.discovery.messages.query](#)), [method](#), 168
[178](#)
[query\(\)](#) ([aries_cloudagent.messaging.models.base_record.BaseRecord](#) [class method](#)), 50
[Query.Meta](#) (class in [aries_cloudagent.protocols.discovery.messages.query](#)), [method](#), 200
[178](#)
[query_features\(\)](#) (in [module](#) [aries_cloudagent.protocols.discovery.routes](#)), [method](#), 200
[179](#)
[QueryHandler](#) (class in [aries_cloudagent.protocols.discovery.handlers.query_handler](#)), [method](#), 155
[177](#)
[QueryResultSchema](#) (class in [aries_cloudagent.protocols.discovery.routes](#)), [method](#), 168
[179](#)
[QuerySchema](#) (class in [aries_cloudagent.protocols.discovery.messages.query](#)), [method](#), 200
[178](#)
[QuerySchema.Meta](#) (class in [aries_cloudagent.protocols.discovery.messages.query](#)), [method](#), 221
[178](#)
[queue_message\(\)](#) ([aries_cloudagent.core.dispatcher.Dispatcher](#) [method](#)), 20
[queue_outbound\(\)](#) ([aries_cloudagent.core.dispatcher.Dispatcher](#) [method](#)), 19
[queued_message_count](#) ([aries_cloudagent.messaging.decorators.transport_decorator.TransportDecoratorSchema](#) [attribute](#)), 47
[QueuedMessage](#) (class in [aries_cloudagent.transport.inbound.delivery_queue](#)), [method](#), 200
[73](#)
[QueuedOutboundMessage](#) (class in [aries_cloudagent.transport.outbound.manager](#)), [method](#), 221
[83](#)
[random_seed\(\)](#) (in [module](#) [aries_cloudagent.wallet.crypto](#)), 104
[receive_credential\(\)](#) ([aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager](#) [method](#)), 168
[receive_credential_ack\(\)](#) ([aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager](#) [method](#)), 168
[receive_inbound\(\)](#) ([aries_cloudagent.transport.inbound.session.InboundSession](#) [method](#)), 79
[receive_invitation\(\)](#) ([aries_cloudagent.protocols.connections.manager.ConnectionManager](#) [method](#)), 155
[receive_offer\(\)](#) ([aries_cloudagent.protocols.credentials.manager.CredentialManager](#) [method](#)), 168
[receive_offer\(\)](#) ([aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager](#) [method](#)), 200
[receive_presentation\(\)](#) ([aries_cloudagent.protocols.present_proof.v1_0.manager.PresentProofManager](#) [method](#)), 221
[receive_presentation_ack\(\)](#) ([aries_cloudagent.protocols.present_proof.v1_0.manager.PresentProofManager](#) [method](#)), 221
[receive_proposal\(\)](#) ([aries_cloudagent.protocols.present_proof.v1_0.manager.PresentProofManager](#) [method](#)), 221
[receive_request\(\)](#) ([aries_cloudagent.protocols.connections.manager.ConnectionManager](#) [method](#)), 155
[receive_request\(\)](#) ([aries_cloudagent.protocols.credentials.manager.CredentialManager](#) [method](#)), 168
[receive_request\(\)](#) ([aries_cloudagent.protocols.present_proof.v1_0.manager.PresentProofManager](#) [method](#)), 221

(aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager
 method), 200
 receive_request() (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentProofManager
 method), 221
 receive_request() (aries_cloudagent.protocols.presentations.manager.PresentationManager
 method), 233
 received_orders (aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator
 attribute), 44
 received_orders (aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecoratorSchema
 attribute), 44
 recip_keys (aries_cloudagent.connections.models.diddoc.Service attribute), 250
 recip_keys (aries_cloudagent.connections.models.diddoc.service.Service attribute), 118
 recip_keys (aries_cloudagent.connections.models.diddoc.service.Service attribute), 122
 recipient_did (aries_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 77
 recipient_did_public (aries_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 77
 recipient_key (aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResultSchema
 attribute), 247
 recipient_key (aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResultSchema
 attribute), 248
 recipient_key (aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResultSchema
 attribute), 249
 recipient_key (aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResultSchema
 attribute), 250
 recipient_keys (aries_cloudagent.connections.models.connection_record.ConnectionRecord attribute), 130
 recipient_keys (aries_cloudagent.connections.models.connection_record.ConnectionRecord attribute), 147
 recipient_verkey (aries_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 77
 record_id (aries_cloudagent.protocols.routing.models.route_record.RouteRecordSchema attribute), 248
 RECORD_ID_NAME (aries_cloudagent.connections.models.connection_record.ConnectionRecord attribute), 126
 RECORD_ID_NAME (aries_cloudagent.messaging.models.base_record.BaseRecord attribute), 49
 RECORD_ID_NAME (aries_cloudagent.protocols.credentials.models.credential_definition.CredentialDefinition attribute), 166
 RECORD_ID_NAME (aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager attribute), 197
 RECORD_ID_NAME (aries_cloudagent.protocols.present_proof.v1_0.manager.PresentProofManager attribute), 218
 RECORD_ID_NAME (aries_cloudagent.protocols.presentations.models.presentation.Presentation attribute), 232
 record_tags (aries_cloudagent.messaging.models.base_record.BaseRecord attribute), 50
 RECORD_TYPE (aries_cloudagent.connections.models.connection_record.ConnectionRecord attribute), 126
 RECORD_TYPE (aries_cloudagent.messaging.models.base_record.BaseRecord attribute), 50

register()	(in module <i>aries_cloudagent.messaging.schemas.routes</i>), 54	register_nym() (<i>aries_cloudagent.ledger.base.BaseLedger</i> method), 26
register()	(in module <i>aries_cloudagent.protocols.actionmenu.routes</i>), 141	register_nym() (<i>aries_cloudagent.ledger.indy.IndyLedger</i> method), 29
register()	(in module <i>aries_cloudagent.protocols.basicmessage.routes</i>), 144	register_package() (<i>aries_cloudagent.core.plugin_registry.PluginRegistry</i> method), 22
register()	(in module <i>aries_cloudagent.protocols.connections.routes</i>), 159	register_plugin() (<i>aries_cloudagent.core.plugin_registry.PluginRegistry</i> method), 22
register()	(in module <i>aries_cloudagent.protocols.credentials.routes</i>), 176	register_transport() (<i>aries_cloudagent.transport.inbound.manager.InboundTransportManager</i> method), 75
register()	(in module <i>aries_cloudagent.protocols.discovery.routes</i>), 179	release() (<i>aries_cloudagent.cache.base.BaseCache</i> method), 6
register()	(in module <i>aries_cloudagent.protocols.introduction.routes</i>), 185	release() (<i>aries_cloudagent.cache.base.CacheKeyLock</i> method), 7
register()	(in module <i>aries_cloudagent.protocols.issue_credential.routes</i>), 207	remove() (<i>aries_cloudagent.wallet.indy.IndyWallet</i> method), 108
register()	(in module <i>aries_cloudagent.protocols.issue_credential.v1_0.routes</i>), 206	remove_field() (<i>aries_cloudagent.messaging.decorators.base.BaseDecorator</i> method), 39
register()	(in module <i>aries_cloudagent.protocols.present_proof.routes</i>), 228	remove_keys_for_did() (<i>aries_cloudagent.protocols.connections.manager.ConnectionManager</i> method), 155
register()	(in module <i>aries_cloudagent.protocols.present_proof.v1_0.routes</i>), 227	remove_message_for_key() (<i>aries_cloudagent.transport.inbound.delivery_queue.DeliveryQueue</i> method), 73
register()	(in module <i>aries_cloudagent.protocols.presentations.routes</i>), 237	remove_model() (<i>aries_cloudagent.messaging.decorators.base.BaseDecorator</i> method), 39
register()	(in module <i>aries_cloudagent.protocols.trustping.routes</i>), 254	remove_skipped_values() (<i>aries_cloudagent.messaging.models.base.BaseModelSchema</i> method), 48
register()	(in module <i>aries_cloudagent.wallet.routes</i>), 111	remove_webhook_target() (<i>aries_cloudagent.admin.base_server.BaseAdminServer</i> method), 3
register_admin_routes()	(<i>aries_cloudagent.core.plugin_registry.PluginRegistry</i> method), 22	remove_webhook_target() (<i>aries_cloudagent.admin.server.AdminServer</i> method), 5
register_class()	(<i>aries_cloudagent.transport.outbound.manager.OutboundTransportManager</i> method), 83	RepeatAttempt (class in <i>aries_cloudagent.utils.repeat</i>), 91
register_controllers()	(<i>aries_cloudagent.core.protocol_registry.ProtocolRegistry</i> method), 23	RepeatSequence (class in <i>aries_cloudagent.utils.repeat</i>), 91
register_ledger_nym()	(in module <i>aries_cloudagent.ledger.routes</i>), 31	replace_local_did_metadata() (<i>aries_cloudagent.wallet.base.BaseWallet</i> method), 98
register_message_types()	(<i>aries_cloudagent.core.protocol_registry.ProtocolRegistry</i> method), 23	replace_local_did_metadata() (<i>aries_cloudagent.wallet.basic.BasicWallet</i> method), 100
		replace_local_did_metadata() (<i>aries_cloudagent.wallet.indy.IndyWallet</i> method), 108
		replace_signatures() (<i>aries_cloudagent.messaging.agent_message.AgentMessageSchema</i> method), 57

replace_signing_key_metadata()	(aries_cloudagent.wallet.base.BaseWallet method), 98	resolve_class()	(in module aries_cloudagent.messaging.models.base), 48
replace_signing_key_metadata()	(aries_cloudagent.wallet.basic.BasicWallet method), 101	resolve_inbound_connection()	(aries_cloudagent.protocols.connections.manager.ConnectionManager method), 155
replace_signing_key_metadata()	(aries_cloudagent.wallet.indy.IndyWallet method), 108	resolve_message_class()	(aries_cloudagent.core.protocol_registry.ProtocolRegistry method), 23
reply_mode(aries_cloudagent.transport.inbound.session.InboundSession attribute), 79		schema_property()	(in module aries_cloudagent.messaging.models.base), 48
REPLY_MODE_ALL(aries_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 76		resource()	(in module aries_cloudagent.connections.models.diddoc.util), 123
REPLY_MODE_NONE(aries_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 76		responder(aries_cloudagent.protocols.introduction.messages.invitation.Invitation attribute), 183	
REPLY_MODE_THREAD(aries_cloudagent.transport.inbound.receipt.MessageReceipt attribute), 76		ResponderError, 60	
reply_thread_ids(aries_cloudagent.transport.inbound.sessions.InboundSession attribute), 79		session_inbound_session(aries_cloudagent.transport.inbound.session.InboundSession attribute), 79	
reply_verkeys(aries_cloudagent.transport.inbound.session.InboundSession attribute), 79		RESPONSE_NOT_ACCEPTED	
request(aries_cloudagent.protocols.credentials.messages.credentials.messages.problem_report.ProblemReport attribute), 163		request(aries_cloudagent.protocols.credentials.messages.credentials.messages.problem_report.ProblemReport attribute), 149	
request(aries_cloudagent.protocols.presentations.messages.presentations.messages.problem_report.ProblemReport attribute), 230		request_schema(aries_cloudagent.protocols.connections.messages.problem_report.ProblemReport attribute), 149	
request_end() (aries_cloudagent.transport.stats.StatsTracer method), 88		response_requested	
request_id(aries_cloudagent.connections.models.connection_record.ConnectionRecord attribute), 128		result(aries_cloudagent.admin.server.AdminModulesSchema attribute), 149	
REQUEST_NOT_ACCEPTED(aries_cloudagent.protocols.connections.messages.problem_report.ProblemReport attribute), 149		result(aries_cloudagent.cache.base.CacheKeyLock attribute), 7	
request_presentations_attach(aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof.v1_0.messages.problem_report.ProblemReport attribute), 216		route_updated.RouteUpdated attribute), 250	
REQUEST_PROCESSING_ERROR(aries_cloudagent.protocols.connections.messages.problem_report.ProblemReport attribute), 149		RESULT_CLIENT_ERROR	
request_start() (aries_cloudagent.transport.stats.StatsTracer method), 88		RESULT_NO_CHANGE(aries_cloudagent.protocols.routing.models.route_updated.RouteUpdated attribute), 249	
RequestContext (class in aries_cloudagent.messaging.request_context), 58		RESULT_SERVER_ERROR(aries_cloudagent.protocols.routing.models.route_updated.RouteUpdated attribute), 249	
requests_attach(aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential.v1_0.messages.problem_report.ProblemReport attribute), 194		route_updated.RouteUpdated attribute), 249	
required(aries_cloudagent.protocols.actionmenu.models.actionmenu.models.problem_report.ProblemReport attribute), 137		route_updated.RouteUpdated attribute), 249	
reset() (aries_cloudagent.transport.queue.base.BaseMessageQueue method), 85		route_updated.RouteUpdated attribute), 249	
reset() (aries_cloudagent.transport.queue.basic.BasicMessageQueue method), 86		route_updated.RouteUpdated attribute), 249	
reset() (aries_cloudagent.utils.stats.Collector method), 92		route_updated.RouteUpdated attribute), 249	

RouteRecordSchema (class in aries_cloudagent.protocols.routing.messages.route_update_response, 248
RouteRecordSchema.Meta (class in aries_cloudagent.protocols.routing.models.route_update), 248
RouteUpdateSchema (class in aries_cloudagent.protocols.routing.messages.route_update_response, 248
RouteUpdateSchema.Meta (class in aries_cloudagent.protocols.routing.models.route_update), 248
routes (aries_cloudagent.protocols.routing.messages.route_query_response, 243
RouteUpdate (class in routing_keys (aries_cloudagent.connections.models.connection_target), 130
RouteUpdate.Meta (class in routing_keys (aries_cloudagent.connections.models.diddoc.Service), 118
RouteUpdated (class in routing_keys (aries_cloudagent.connections.messages.connection_update), 147
RouteUpdated.Meta (class in routing_state (aries_cloudagent.connections.models.connection_record), 128
RouteUpdatedSchema (class in routing_state (aries_cloudagent.connections.models.connection_record), 126
RouteUpdatedSchema.Meta (class in routing_state (aries_cloudagent.connections.models.connection_record), 126
RouteUpdateRequest (class in routing_state (aries_cloudagent.connections.models.connection_record), 126
RouteUpdateRequest.Meta (class in routing_state (aries_cloudagent.connections.models.connection_record), 126
RouteUpdateRequestHandler (class in routing_state (aries_cloudagent.connections.models.connection_record), 250
RouteUpdateRequestSchema (class in routing_state (aries_cloudagent.connections.models.connection_record), 121
RouteUpdateRequestSchema.Meta (class in routing_state (aries_cloudagent.connections.models.connection_record), 117
RouteUpdateResponse (class in run () (aries_cloudagent.utils.task_queue.TaskQueue), 94
RouteUpdateResponse.Meta (class in run_command () (in module aries_cloudagent.commands), 8
RouteUpdateResponseHandler (class in save () (aries_cloudagent.messaging.models.base_record.BaseRecord), 51
RouteUpdateResponseSchema (class in save_connection_menu () (in module aries_cloudagent.protocols.actionmenu.util), 244
RouteUpdateResponseSchema.Meta (class in save_connection_menu () (in module aries_cloudagent.protocols.actionmenu.util), 244

141	schema_class (aries_cloudagent.protocols.connections.models.connect
scan_subpackages ()	attribute), 150
(aries_cloudagent.utils.classloader.ClassLoader	schema_class (aries_cloudagent.protocols.credentials.messages.creden
class method), 90	attribute), 161
Schema (aries_cloudagent.messaging.models.base.BaseModel	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 47	attribute), 162
schema_class (aries_cloudagent.connections.models.connection_re	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 126	attribute), 163
schema_class (aries_cloudagent.connections.models.connection_re	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 129	attribute), 163
schema_class (aries_cloudagent.messaging.ack.messages.Ack	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 32	attribute), 166
schema_class (aries_cloudagent.messaging.agent_message.AgentMessage	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 55	attribute), 177
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 36	attribute), 178
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 37	attribute), 181
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 40	attribute), 182
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 41	attribute), 183
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 42	attribute), 189
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 44	attribute), 190
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 45	attribute), 191
schema_class (aries_cloudagent.messaging.decorators.attach_decorator	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 46	attribute), 193
schema_class (aries_cloudagent.messaging.models.base.BaseModel	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 47	attribute), 194
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 133	attribute), 187
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 133	attribute), 188
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 134	attribute), 197
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 135	attribute), 209
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 136	attribute), 211
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 137	attribute), 210
schema_class (aries_cloudagent.protocols.actionmenu.messages.men	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 143	attribute), 213
schema_class (aries_cloudagent.protocols.connections.messages.con	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 146	attribute), 214
schema_class (aries_cloudagent.protocols.connections.messages.con	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 148	attribute), 215
schema_class (aries_cloudagent.protocols.connections.messages.con	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 148	attribute), 215
schema_class (aries_cloudagent.protocols.connections.messages.con	schema_class (aries_cloudagent.protocols.credentials.messages.creden
attribute), 149	attribute), 218


```

schema_class(aries_cloudagent.protocols.presentations.messages.presentation.request.PresentationRequest.Meta
attribute), 229
schema_class(aries_cloudagent.protocols.presentations.messages.presentation.request.PresentationRequest.Meta
attribute), 230
schema_class(aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange.Meta
attribute), 232
schema_class(aries_cloudagent.protocols.problem_report.messages.problem_report.ProblemReport.Meta
attribute), 239
schema_class(aries_cloudagent.protocols.routing.messages.forward_forwards.ForwardMeta
attribute), 241
schema_class(aries_cloudagent.protocols.routing.messages.route_query_request.RouteQueryRequest.Meta
attribute), 242
schema_class(aries_cloudagent.protocols.routing.messages.route_query_response.RouteQueryResponse.Meta
attribute), 243
schema_class(aries_cloudagent.protocols.routing.messages.route_update_request.RouteUpdateRequest.Meta
attribute), 244
schema_class(aries_cloudagent.protocols.routing.messages.route_update_response.RouteUpdateResponse.Meta
attribute), 244
schema_class(aries_cloudagent.protocols.routing.models.pagination_pagination.Meta
attribute), 245
schema_class(aries_cloudagent.protocols.routing.models.pagination_pagination.Meta
attribute), 246
schema_class(aries_cloudagent.protocols.routing.models.route_query_result.RouteQueryResult.Meta
attribute), 247
schema_class(aries_cloudagent.protocols.routing.models.route_record.RouteRecord.Meta
attribute), 248
schema_class(aries_cloudagent.protocols.routing.models.route_update.RouteUpdate.Meta
attribute), 248
schema_class(aries_cloudagent.protocols.routing.models.route_updated.RouteUpdated.Meta
attribute), 249
schema_class(aries_cloudagent.protocols.trustping.messages.ping.Ping.Meta
attribute), 252
schema_class(aries_cloudagent.protocols.trustping.messages.ping_response.PingResponse.Meta
attribute), 253
schema_id(aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema
attribute), 167
schema_id(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposalSchema
attribute), 193
schema_id(aries_cloudagent.protocols.issue_credential.v1_0.model.credential_exchange_v1_0.CredentialExchangeSchema
attribute), 198
schema_issuer_did
(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposalSchema
attribute), 193
schema_name(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposalSchema
attribute), 193
schema_version(aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal.CredentialProposalSchema
attribute), 193
SchemaGetResultsSchema (class in (aries_cloudagent.messaging.schemas.routes),
52
schemas_created() (in module (aries_cloudagent.messaging.schemas.routes),
54
schemas_get_schema() (in module

```

`send_outbound()` (`aries_cloudagent.core.dispatcher.DispatcherResponse` (class in `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`), 21
`method`), 21
`send_outbound()` (`aries_cloudagent.messaging.responder.BaseResponder` (class in `aries_cloudagent.messaging.models.base.BaseModel`), 60
`method`), 60
`send_outbound()` (`aries_cloudagent.messaging.responder.MockResponder` (class in `aries_cloudagent.connections.models.diddoc.DIDDoc`), 60
`method`), 60
`send_presentation_ack()` (`aries_cloudagent.protocols.present_proof.v1_0.manager.PresentationManager` (class in `aries_cloudagent.connections.models.diddoc.DIDDoc`), 221
`method`), 221
`send_reply()` (`aries_cloudagent.messaging.responder.BaseResponder` (class in `aries_cloudagent.connections.models.diddoc.DIDDoc`), 60
`method`), 60
`send_reply()` (`aries_cloudagent.messaging.responder.MockResponder` (class in `aries_cloudagent.connections.models.diddoc.DIDDoc`), 60
`method`), 60
`send_schema()` (`aries_cloudagent.ledger.indy.IndyLedger` (class in `aries_cloudagent.protocols.actionmenu.base_service.BaseMenuService`), 29
`method`), 29
`send_webhook()` (`aries_cloudagent.admin.base_server.BaseAdminServer` (class in `aries_cloudagent.protocols.introduction.base_service.BaseIntroductionService`), 3
`method`), 3
`send_webhook()` (`aries_cloudagent.admin.server.AdminResponder` (class in `aries_cloudagent.cache.base.BaseCache`), 4
`method`), 4
`send_webhook()` (`aries_cloudagent.admin.server.AdminServer` (class in `aries_cloudagent.cache.basic.BasicCache`), 5
`method`), 5
`send_webhook()` (`aries_cloudagent.core.dispatcher.DispatcherResponse` (class in `aries_cloudagent.connections.models.diddoc.DIDDoc`), 7
`method`), 7
`send_webhook()` (`aries_cloudagent.messaging.models.base_record.BaseRecord` (class in `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`), 51
`method`), 51
`send_webhook()` (`aries_cloudagent.messaging.responder.BaseResponder` (class in `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`), 60
`method`), 60
`send_webhook()` (`aries_cloudagent.messaging.responder.MockResponder` (class in `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`), 60
`method`), 60
`send_webhook()` (`aries_cloudagent.protocols.actionmenu.driver.AriesDriverMenuService` (class in `aries_cloudagent.ledger.indy.IndyWallet`), 139
`method`), 139
`sender_did()` (`aries_cloudagent.transport.inbound.receipt.MessageReceipt` (class in `aries_cloudagent.config.settings.Settings`), 77
`attribute`), 77
`sender_key()` (`aries_cloudagent.connections.models.connection_target.ConnectionTarget` (class in `aries_cloudagent.wallet.base.BaseWallet`), 130
`attribute`), 130
`sender_order()` (`aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator` (class in `aries_cloudagent.transport.inbound.session.InboundSession`), 44
`attribute`), 44
`sender_order()` (`aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator` (class in `aries_cloudagent.transport.inbound.session.InboundSession`), 44
`attribute`), 44
`sender_verkey()` (`aries_cloudagent.transport.inbound.receipt.MessageReceipt` (class in `aries_cloudagent.messaging.agent_message.AgentMessage`), 77
`attribute`), 77
`SendMenuSchema` (class in `aries_cloudagent.protocols.actionmenu.routes`), 140
`set_urlsafes_b64()` (in `aries_cloudagent.wallet.util`), 112
`set_value()` (`aries_cloudagent.config.settings.Settings` (class in `aries_cloudagent.wallet.routes`), 18
`method`), 18
`SendMessageSchema` (class in `aries_cloudagent.protocols.basicmessage.routes`), 143
`SetTagPolicyRequestSchema` (class in `aries_cloudagent.wallet.routes`), 111
`SendPresentationRequestSchema` (class in `aries_cloudagent.protocols.presentations.routes`), 235
`settings` (`aries_cloudagent.config.injection_context.InjectionContext` (class in `aries_cloudagent.config.injector.Injector`), 14
`attribute`), 14
`sent_time()` (`aries_cloudagent.protocols.basicmessage.messages.basicmessage.BasicMessageSchema` (class in `aries_cloudagent.config.settings`), 143
`attribute`), 143
`serialize()` (`aries_cloudagent.connections.models.diddoc.DIDDoc` (class in `aries_cloudagent.config.settings`), 116
`method`), 116
`SettingsError`, 12

`setup()` (`aries_cloudagent.core.conductor.Conductor` method), 19
`setup()` (`aries_cloudagent.core.dispatcher.Dispatcher` method), 21
`setup()` (`aries_cloudagent.transport.inbound.manager.InboundTransportManager` method), 75
`setup()` (`aries_cloudagent.transport.outbound.manager.OutboundTransportManager` method), 83
`sha256` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 37
`sha256_` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 38
`SHA256Hash` (class in `socket_connect_start()` (`aries_cloudagent.messaging.valid`), 63
`shutdown_app()` (in module `aries_cloudagent.commands.start`), 8
`sig` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 37
`sig_` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 38
`sig_data` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 43
`sign()` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` method), 38
`sign_field()` (`aries_cloudagent.messaging.agent_message.AgentMessage` method), 56
`sign_message()` (`aries_cloudagent.wallet.base.BaseWallet` method), 98
`sign_message()` (`aries_cloudagent.wallet.basic.BasicWallet` method), 101
`sign_message()` (`aries_cloudagent.wallet.indy.IndyWallet` method), 109
`sign_message()` (in module `aries_cloudagent.wallet.crypto`), 104
`sign_pk_from_sk()` (in module `aries_cloudagent.wallet.crypto`), 105
`signature` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 43
`signature_type` (`aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator` attribute), 43
`SignatureDecorator` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 42
`SignatureDecorator.Meta` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 42
`SignatureDecoratorSchema` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 43
`SignatureDecoratorSchema.Meta` (class in `aries_cloudagent.messaging.decorators.signature_decorator`), 43
`signatures` (`aries_cloudagent.messaging.decorators.attach_decorator.AttachDecorator` attribute), 38

(aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10PresentationExchange
attribute), 218 stop () (aries_cloudagent.transport.outbound.http.HttpTransport
method), 81
STATE_REQUEST_RECEIVED (aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange
attribute), 232 stop () (aries_cloudagent.transport.outbound.ws.WsTransport
method), 83
STATE_REQUEST_SENT (aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange
attribute), 166 stop () (aries_cloudagent.transport.queue.base.BaseMessageQueue
method), 85
STATE_REQUEST_SENT (aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange.V10CredentialExchange
attribute), 197 stop () (aries_cloudagent.transport.queue.base.BaseMessageQueue
method), 86
STATE_REQUEST_SENT (aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10PresentationExchange
attribute), 218 stop () (aries_cloudagent.utils.stats.Timer method), 93
STATE_REQUEST_SENT (aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10PresentationExchange
attribute), 218 stop () (aries_cloudagent.utils.stats.Timer method), 93
STATE_REQUEST_SENT (aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange
attribute), 232 STORAGE_TYPES (aries_cloudagent.storage.provider.StorageProvider
attribute), 70 StorageDuplicateError, 67
STATE_RESPONSE (aries_cloudagent.connections.models.connection_record.ConnectionRecord
attribute), 126 StorageNotFoundError, 68
STATE_RETRY (aries_cloudagent.transport.outbound.manager.OutboundMessageQueue (class in
attribute), 83 aries_cloudagent.storage.provider), 70
STATE_STORED (aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange (class in
attribute), 166 aries_cloudagent.storage.record), 70
STATE_VERIFIED (aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10PresentationExchange
attribute), 218 store (aries_cloudagent.storage.base.BaseStorageRecordSearch
method), 68
STATE_VERIFIED (aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange
attribute), 232 store_credential ()
Stats (class in aries_cloudagent.utils.stats), 92 (aries_cloudagent.holder.indy.IndyHolder
method), 24
StatsProvider (class in aries_cloudagent.config.provider), 17 store_credential ()
StatsTracer (class in aries_cloudagent.transport.stats), 87 (aries_cloudagent.protocols.credentials.manager.CredentialManager
method), 168
status (aries_cloudagent.messaging.ack.message.AckSchema (class in aries_cloudagent.messaging.ack.message), 32 store_credential ()
(aries_cloudagent.protocols.issue_credential.v1_0.manager.CredentialManager
method), 197
status_handler () (aries_cloudagent.admin.server.AdminServer method), 200
method), 5 store_document ()
status_reset_handler () (aries_cloudagent.admin.server.AdminServer method), 155
method), 5 str_to_b64 () (in module
aries_cloudagent.wallet.util), 112
stop () (aries_cloudagent.admin.base_server.BaseAdminServer method), 3 str_to_datetime () (in module
aries_cloudagent.messaging.util), 61
stop () (aries_cloudagent.admin.server.AdminServer method), 5 str_to_epoch () (in module
aries_cloudagent.messaging.util), 61
stop () (aries_cloudagent.core.conductor.Conductor method), 19 strip_tag_prefix ()
stop () (aries_cloudagent.transport.inbound.base.BaseInboundTransport method), 71 (aries_cloudagent.messaging.models.base_record.BaseRecord
class method), 52
stop () (aries_cloudagent.transport.inbound.http.HttpTransport method), 74 submit_label (aries_cloudagent.protocols.actionmenu.models.menu_item.MenuItem
attribute), 135
stop () (aries_cloudagent.transport.inbound.manager.InboundTransportManager method), 75
stop () (aries_cloudagent.transport.inbound.ws.WsTransport method), 80 sha_digest () (aries_cloudagent.ledger.base.BaseLedger
method), 26
stop () (aries_cloudagent.transport.outbound.base.BaseOutboundTransport

`taa_digest()` (`aries_cloudagent.ledger.indy.IndyLedger` attribute), 198
`taa_rough_timestamp()` (`aries_cloudagent.ledger.indy.IndyLedger` method), 29
`taa_rough_timestamp()` (`aries_cloudagent.ledger.indy.IndyLedger` method), 29
`TAAAcceptanceSchema` (class in `aries_cloudagent.ledger.routes`), 30
`TAAAcceptSchema` (class in `aries_cloudagent.ledger.routes`), 30
`TAAInfoSchema` (class in `aries_cloudagent.ledger.routes`), 30
`TAARecordSchema` (class in `aries_cloudagent.ledger.routes`), 31
`TAAResultSchema` (class in `aries_cloudagent.ledger.routes`), 31
`TAG_NAMES` (`aries_cloudagent.connections.models.connection_record.ConnectionRecord` attribute), 126
`TAG_NAMES` (`aries_cloudagent.messaging.models.base_record.BaseRecord` attribute), 49
`TAG_NAMES` (`aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchange` attribute), 166
`TAG_NAMES` (`aries_cloudagent.protocols.issue_credential.v1_0.models.issue_credential_v1_0_protocol.ExchangePort` attribute), 197
`TAG_NAMES` (`aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange.V10PresentationExchange` attribute), 218
`TAG_NAMES` (`aries_cloudagent.protocols.presentations.models.presentation_exchange.PresentationExchange` attribute), 232
`tag_query` (`aries_cloudagent.storage.base.BaseStorageRecordSchema` attribute), 65
`tags` (`aries_cloudagent.messaging.models.base_record.BaseRecord` attribute), 52
`task` (`aries_cloudagent.utils.task_queue.PendingTask` attribute), 93
`task_done()` (`aries_cloudagent.transport.queue.base.BaseMessageQueue` method), 85
`task_done()` (`aries_cloudagent.transport.queue.basic.BasicMessageQueue` method), 86
`task_exc_info()` (in module `aries_cloudagent.utils.task_queue`), 95
`TaskQueue` (class in `aries_cloudagent.utils.task_queue`), 93
`their_id` (`aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema` attribute), 128
`their_label` (`aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema` attribute), 128
`their_role` (`aries_cloudagent.connections.models.connection_record.ConnectionRecordSchema` attribute), 128
`thid` (`aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator` attribute), 44
`thid` (`aries_cloudagent.messaging.decorators.thread_decorator.ThreadDecorator` attribute), 44
`thread_id` (`aries_cloudagent.protocols.credentials.models.credential_exchange.CredentialExchangeSchema` attribute), 167
`thread_id` (`aries_cloudagent.protocols.issue_credential.v1_0.models.issue_credential_v1_0_protocol.ExchangePort` attribute), 167

[to_dict\(\) \(aries_cloudagent.connections.models.diddoc.PublicKey.PublicKey method\), 121](#)
[to_dict\(\) \(aries_cloudagent.connections.models.diddoc.Service method\), 118](#)
[to_dict\(\) \(aries_cloudagent.connections.models.diddoc.service.Service method\), 123](#)
[to_dict\(\) \(aries_cloudagent.messaging.decorators.base.BaseDecoratorSet method\), 39](#)
[to_json\(\) \(aries_cloudagent.connections.models.diddoc.DIDDoc method\), 116](#)
[to_json\(\) \(aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc method\), 120](#)
[to_json\(\) \(aries_cloudagent.messaging.models.base.BaseModel method\), 47](#)
[to_url\(\) \(aries_cloudagent.protocols.connections.messages.connection_invitation.ConnectionInvitation method\), 146](#)
[topic_filter \(aries_cloudagent.admin.server.WebhookTarget attribute\), 5](#)
[total \(aries_cloudagent.protocols.routing.models.paginated.PaginatedSchema attribute\), 246](#)
[tracking_uri \(aries_cloudagent.protocols.problem_report.message.ProblemReportSchema attribute\), 239](#)
[TransportDecorator \(class in aries_cloudagent.messaging.decorators.transport_decorator\), 46](#)
[TransportDecorator.Meta \(class in aries_cloudagent.messaging.decorators.transport_decorator\), 46](#)
[TransportDecoratorSchema \(class in aries_cloudagent.messaging.decorators.transport_decorator\), 46](#)
[TransportDecoratorSchema.Meta \(class in aries_cloudagent.messaging.decorators.transport_decorator\), 46](#)
[TransportError, 86](#)
[TransportGroup \(class in aries_cloudagent.config argparse\), 10](#)
[type \(aries_cloudagent.connections.models.diddoc.PublicKey attribute\), 117](#)
[type \(aries_cloudagent.connections.models.diddoc.publickey.PublicKey attribute\), 121](#)
[type \(aries_cloudagent.connections.models.diddoc.Service attribute\), 118](#)
[type \(aries_cloudagent.connections.models.diddoc.service.Service attribute\), 123](#)
[type \(aries_cloudagent.wallet.base.BaseWallet attribute\), 98](#)
[TYPE_ED25519SHA512 \(aries_cloudagent.messaging.decorators.signature_decorator.SignatureDecorator attribute\), 42](#)
[type_filter \(aries_cloudagent.storage.base.BaseStorageRecordSearch attribute\), 65](#)
[unpack\(\) \(aries_cloudagent.transport.pack_format.PackWireFormat method\), 87](#)
[unpack_message\(\) \(aries_cloudagent.wallet.base.BaseWallet method\), 98](#)
[unpack_message\(\) \(aries_cloudagent.wallet.basic.BasicWallet method\), 101](#)
[unpack_message\(\) \(aries_cloudagent.wallet.indy.IndyWallet method\), 109](#)
[unpad\(\) \(in module aries_cloudagent.wallet.util\), 112](#)
[UNREVEALED_CLAIM \(aries_cloudagent.protocols.present_proof.v1_0.message attribute\), 209](#)
[update_endpoint_for_did\(\) \(aries_cloudagent.ledger.base.BaseLedger method\), 29](#)
[update_endpoint_for_did\(\) \(aries_cloudagent.ledger.indy.IndyLedger method\), 29](#)
[update_inbound\(\) \(aries_cloudagent.protocols.connections.manager.ConnectionManager method\), 155](#)
[update_record_tags\(\) \(aries_cloudagent.storage.base.BaseStorage method\), 64](#)
[update_record_tags\(\) \(aries_cloudagent.storage.basic.BasicStorage method\), 66](#)
[update_record_tags\(\) \(aries_cloudagent.storage.indy.IndyStorage method\), 69](#)
[update_record_value\(\) \(aries_cloudagent.storage.base.BaseStorage method\), 64](#)
[update_record_value\(\) \(aries_cloudagent.storage.basic.BasicStorage method\), 66](#)
[update_record_value\(\) \(aries_cloudagent.storage.indy.IndyStorage method\), 69](#)
[update_routes\(\) \(aries_cloudagent.protocols.routing.manager.RoutingManager method\), 251](#)
[update_settings\(\) \(aries_cloudagent.config.base_context.ContextBuilder method\), 13](#)
[update_settings\(\) \(aries_cloudagent.config.injection_context.InjectionContext method\), 14](#)
[updated \(aries_cloudagent.protocols.routing.messages.route_update_request.RouteUpdateRequest attribute\), 245](#)
[updated_at \(aries_cloudagent.messaging.models.base_record.BaseRecord attribute\), 52](#)
[updated_at \(aries_cloudagent.protocols.routing.models.route_record.RouteRecord attribute\), 248](#)
[updates \(aries_cloudagent.protocols.routing.messages.route_update_request.RouteUpdateRequest attribute\), 244](#)

[UUIDFour \(class in aries_cloudagent.messaging.valid\)](#), 218
[63](#)

V

[V10AttributeMimeTypesResultSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 201
[V10CredentialExchange \(class in aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange\)](#), 195
[V10CredentialExchange.Meta \(class in aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange\)](#), 197
[V10CredentialExchangeListResultSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 201
[V10CredentialExchangeSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange\)](#), 197
[V10CredentialExchangeSchema.Meta \(class in aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange\)](#), 197
[V10CredentialIssueRequestSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 201
[V10CredentialOfferRequestSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 202
[V10CredentialProblemReportRequestSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 202
[V10CredentialProposalRequestMandSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 202
[V10CredentialProposalRequestOptSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 203
[V10CredentialProposalRequestSchemaBase \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 203
[V10CredentialStoreRequestSchema \(class in aries_cloudagent.protocols.issue_credential.v1_0.routes\)](#), 203
[V10PresentationExchange \(class in aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange\)](#), 216
[V10PresentationExchange.Meta \(class in aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange\)](#), 218
[V10PresentationExchangeListSchema \(class in aries_cloudagent.protocols.present_proof.v1_0.routes\)](#), 224
[V10PresentationExchangeSchema \(class in aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange\)](#), 221

(aries_cloudagent.protocols.presentations.manager.PresentationManager (class in
 method), 233
 verify_presentation() (aries_cloudagent.verifier.indy.IndyVerifier (class in
 method), 95
 verify_signatures() (aries_cloudagent.messaging.agent_message.AgentMessage (class in
 method), 56
 verify_signed_field() (aries_cloudagent.messaging.agent_message.AgentMessage (class in
 method), 56
 verify_signed_message() (in module aries_cloudagent.wallet.crypto), 105
 verkey (aries_cloudagent.wallet.base.DIDInfo (class in module aries_cloudagent.wallet.base), 99
 verkey (aries_cloudagent.wallet.base.KeyInfo (class in module aries_cloudagent.wallet.base), 99
W
 wait_for() (aries_cloudagent.utils.task_queue.TaskQueue (class in module aries_cloudagent.utils), 95
 wait_response() (aries_cloudagent.transport.inbound.session.InboundSession (class in module aries_cloudagent.transport.inbound.session), 79
 wait_until_time (aries_cloudagent.messaging.decorators.timing_decorator.timing_decorator (class in module aries_cloudagent.messaging.decorators), 46
 wallet (aries_cloudagent.storage.indy.IndyStorage (class in module aries_cloudagent.storage.indy), 69
 wallet_config() (in module aries_cloudagent.config.wallet), 18
 wallet_create_did() (in module aries_cloudagent.wallet.routes), 111
 wallet_did_list() (in module aries_cloudagent.wallet.routes), 111
 wallet_get_public_did() (in module aries_cloudagent.wallet.routes), 111
 wallet_get_tagging_policy() (in module aries_cloudagent.wallet.routes), 111
 wallet_set_public_did() (in module aries_cloudagent.wallet.routes), 112
 wallet_set_tagging_policy() (in module aries_cloudagent.wallet.routes), 112
 WALLET_TYPE (aries_cloudagent.wallet.base.BaseWallet (class in module aries_cloudagent.wallet.base), 96
 WALLET_TYPE (aries_cloudagent.wallet.basic.BasicWallet (class in module aries_cloudagent.wallet.basic), 99
 WALLET_TYPE (aries_cloudagent.wallet.indy.IndyWallet (class in module aries_cloudagent.wallet.indy), 106
 WALLET_TYPES (aries_cloudagent.wallet.provider.WalletProvider (class in module aries_cloudagent.wallet.provider), 110
 WalletDuplicateError, 105
 WalletError, 105
 WalletGroup (class in module aries_cloudagent.config.argsparse), 10
 WalletNotFoundError, 105
 WebhookPayload (aries_cloudagent.messaging.models.base_record.BaseRecord (class in module aries_cloudagent.messaging.models), 52
 WebhookRouter (aries_cloudagent.core.conductor.Conductor (class in module aries_cloudagent.core.conductor), 19
 WEBHOOK_TOPIC (aries_cloudagent.connections.models.connection_record.ConnectionRecord (class in module aries_cloudagent.connections.models), 126
 WEBHOOK_TOPIC (aries_cloudagent.messaging.models.base_record.BaseRecord (class in module aries_cloudagent.messaging.models), 49
 web_hook_topic (aries_cloudagent.messaging.models.base_record.BaseRecord (class in module aries_cloudagent.messaging.models), 52
 WEBHOOK_TOPIC (aries_cloudagent.protocols.credentials.models.credentials_models.credentials_models (class in module aries_cloudagent.protocols.credentials.models), 166
 WEBHOOK_TOPIC (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0.messages (class in module aries_cloudagent.protocols.issue_credential.v1_0.messages), 197
 WEBHOOK_TOPIC (aries_cloudagent.protocols.present_proof.v1_0.messages.present_proof_v1_0.messages (class in module aries_cloudagent.protocols.present_proof.v1_0.messages), 218
 WEBHOOK_TOPIC (aries_cloudagent.protocols.presentations.models.presentations_models.presentations_models (class in module aries_cloudagent.protocols.presentations.models), 232
 WebhookTarget (class in module aries_cloudagent.admin.server), 5
 websocket_handler() (aries_cloudagent.admin.server.AdminServer (class in module aries_cloudagent.admin.server), 5
 where (aries_cloudagent.protocols.problem_report.message.ProblemReport (class in module aries_cloudagent.protocols.problem_report.message), 239
 who_retries (aries_cloudagent.protocols.problem_report.message.ProblemReport (class in module aries_cloudagent.protocols.problem_report.message), 239
 wire_format (aries_cloudagent.transport.outbound.base.BaseOutbound (class in module aries_cloudagent.transport.outbound.base), 80
 WireFormatError, 86
 WitnessSchema (class in module aries_cloudagent.protocols.credentials.routes), 173
 wrap() (aries_cloudagent.utils.stats.Collector (class in module aries_cloudagent.utils.stats), 92
 wrap_coro() (aries_cloudagent.utils.stats.Collector (class in module aries_cloudagent.utils.stats), 92
 wrap_error() (aries_cloudagent.ledger.indy.IndyErrorHandler (class in module aries_cloudagent.ledger.indy), 27
 wrap_fn() (aries_cloudagent.utils.stats.Collector (class in module aries_cloudagent.utils.stats), 92
 wrap_indy_cred_req() (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0.messages (class in module aries_cloudagent.protocols.issue_credential.v1_0.messages), 194
 wrap_indy_credential() (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0.messages (class in module aries_cloudagent.protocols.issue_credential.v1_0.messages), 190
 wrap_indy_offer() (aries_cloudagent.protocols.issue_credential.v1_0.messages.issue_credential_v1_0.messages (class in module aries_cloudagent.protocols.issue_credential.v1_0.messages), 191
 WsTransport (class in module aries_cloudagent.transport.inbound.ws),

79
WsTransport (class in
aries_cloudagent.transport.outbound.ws),
84